I'M ONLY HUMAN: GAME THEORY WITH COMPUTATIONALLY BOUNDED AGENTS

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Lior Ben Zion Seeman

August 2015

I'M ONLY HUMAN:

GAME THEORY WITH COMPUTATIONALLY BOUNDED AGENTS

Lior Ben Zion Seeman, Ph.D.

Cornell University 2015

Rationality in games and decisions is traditionally understood as requiring that agents act optimally. However, as pointed out by Simon [64] acting optimally might be hard. We study how the analysis of games and behaviors changes when agents are assumed to be rational but *computationally bounded*, and thus cannot always act optimally.

We first argue that some observed "irrational" human behaviors can be explained by viewing people as computationally bounded agents. We show that adding computation costs into interactive settings have some unintuitive consequences that might lead to behaviors that seem irrational at first. We then consider a dynamic decision problem, and show that some observed human behavior can be actually explained by modeling people as computationally bounded agents that are doing as well as they can, given their limitations.

We then develop appropriate models of computationally bounded agents modeled as polynomial-time TM. We study the implications of these models and use these models to analyze different aspect of economic systems. We first develop a model appropriate for a repeated game setting and show that problems that are considered intractable, such as computing a NE in repeated games, actually become tractable in this model. We then develop a model of *computational games*, which are finite extensive-form games played by computationally bounded players and show an application of this model to the analyzing of

cryptographic protocols from a game theoretic perspective.

BIOGRAPHICAL SKETCH

Lior Seeman was born on October 13th, 1983 in Netanya, Israel. He received a B.Sc. with a double major in Computer Science and Management from Tel-Aviv University in June 2008 where he graduated summa cum laude. He expects to receive a Ph.D. in Computer Science with a minor in Applied Mathematics from Cornell University in August 2015.

To my loving wife Rotem and my parents Dan and Zipora.

ACKNOWLEDGEMENTS

I first want to thank my advisors, Joe Halpern and Rafael Pass. Their guidance and support throughout my PhD have been invaluable. They encouraged me to tackle interesting and challenging problems, through countless meetings and discussions in which they generously shared their inspiring insights and ideas with me. I would also want to thank them for the trust they had in me and the flexibility they gave me to both pursue other research agendas as well as accommodate personal constraints throughout my PhD.

I am grateful for having the opportunity to be part of the Cornell community. I cannot imagine a more enriching environment for research. I learned so much from all my interactions with the amazing faculty at Cornell. Be it courses, seminars or just random conversations, these interactions frequently left me curious to learn more and filled me with new questions and ideas to pursue. A special thanks for Larry Blume for serving on my committee. I particularly enjoyed my incredible fellow PhD students at Cornell. They have been both inspiring colleagues as well as good friends. It has also been a privilege to spend the last two years as part of the Cornell Tech community. It has been a great fun to see the astonishing growth and accomplishments of this new endeavor and I am truly grateful to have been given the opportunity to be a part of that.

I have been lucky to have many great collaborators throughout my PhD, and I would like to thank all of them. A special thanks to Yaron Singer, for being such a great collaborator, mentor and friend. Thanks for all the hectic hours working together and numerous advices throughout my PhD.

I would also like to thank my parents, Dan and Zipora Seeman, for encouraging me to always pursue my dreams and their support in this process.

Last but definitely not least, I would like to thank my wife, Rotem. Without her constant encouragement I would have never been able to get here. I'd like thank her for the initial push to follow this dream and the endless and unconditional support throughout both the highs and the lows of this long and hard process. I am eternally in debt for her numerous sacrifices to allow me to pursue my dream.

This work was supported in part by a Simons Foundation award for graduate students in theoretical computer science #315783, by NSF grants IIS-0812045, IIS-0911036 and CCF-1214844, by AFOSR grants FA9550-08-1-0438 and FA9550-08-1-0266, by ARO grants W911NF-09-1-0281 and W911NF-14-1-0017, and by the Multidisciplinary University Research Initiative (MURI) program administered by the AFOSR under grant FA9550-12-1-0040.

TABLE OF CONTENTS

LIST OF FIGURES

Part I

Introduction and preliminaries

CHAPTER 1

INTRODUCTION

1.1 Motivations and Goals

How do people decide what to do in various situations? How do they choose the best move out of a set of alternatives? How can we predict what the outcome of an interaction will be? Such questions have been in the heart of both *decision theory*, which is "the theory of *rational* decision making" [57], and of *game theory*, which can be defined as "the study of mathematical models of conflict and cooperation between intelligent *rational* decision-makers" [53].

But what does *rational* mean here? Traditionally, it has been interpreted as saying that players act optimally (*rationality as optimization*) given (their beliefs about) the other players' strategies (and of nature). However, as Simon [64] was the first to point out, agents might have limits on their ability to process information and optimize complex problems (*bounded rationality*), which might "lead to substantial computational simplifications in the making of a choice" [64].

One interesting source of such limits is the computational considerations of the players. These can be both due to inherent computational bounds, so that implementing the optimal strategy requires computational capabilities they do not poses (For example, factoring large numbers) or that such strategies have high computational costs so that it is not worth the effort (For example, exact Bayesian updates of beliefs). Such consideration can have significant consequences when studying game-theoretic problems.

1.1.1 Thesis goal: Study the implications of computationally-bounded agents on the analysis of games and behaviors

Given these considerations, our study of human decision making and strategic interactions should take into account that players are computationally bounded. Moreover, an increasing number of economic systems, from modern ad auctions systems to high-frequency trading, involve strategic interactions by computer programs. Analyses of such systems require models that take the different participants' bounded capabilities into account. Our main goal in this thesis is then:

> *To study how the analysis of games and behaviors changes when agents are rational but computationally bounded, and thus cannot always act optimally?*

This thesis tackles two different aspects of this goal:

- We argue that some observed "irrational" human behaviors can be explained by viewing people as computationally bounded agents that are doing as well as they can, given their limitations.
- We develop appropriate models of computationally bounded agents, study the implications of these models and use these models to analyze different aspect of economic systems.

To accomplish these goals we combine tools and ideas from various fields such as game theory, decision theory, behavioral economics, social science and cryptography.

1.1.2 Approaches for modeling bounded agents

There has been two main approaches for capturing resource-bounded agents in the decision theory and game theory literature. The first aims to capture the intuition that sometimes optimization "is not worth the effort", by charging the agents for the "complexity" of their strategy. This approach can be traced back to Good [23] who pointed out that "we must weigh up the expected time for doing the mathematical and statistical calculations against the expected utility of these calculations". Rubinstein [61] applied this approach to study the repeated prisoners' dilemma game, by assuming the players can only use strategies implemented by a finite automaton and charging them for the number of states in the automaton. (See Kalai [43] for a survey of other related results using this approach.) Ben-Sason, Kalai and Kalai [4] studied a class of games, where, in addition to the payoffs from the game, each player has a cost associated with each action she is playing, regardless of the other players' strategies. These ideas were generalized by Halpern and Pass [27]. In their framework, a player's strategy involves choosing a Turing machine (TM), and the complexity cost of the strategy is a function of the machines chosen by all the players and of the input.

A second approach is to instead of explicitly charging for the complexity of a strategy, to only consider strategies in a set of strategies with bounded complexity. This approach was initiated by Neyman [54], who showed that it can be used to explain cooperation in repeated prisoners' dilemma if the players can only use a finite-automaton of a fixed size. (See [Papadimitriou and Yannakakis 1994] and the references therein for other related results using this approach.) Megiddo and Wigderson [52] considered instead bounded-state

Turing machines to get similar results. Urbano and Vila [66, 67] and Dodis, Halevi and Rabin [14] considered players bounded to strategies implementable by polynomial-time Turing machines and used cryptographic ideas to solve game-theoretic problems.

This thesis addresses questions related to both approaches.

1.2 Contributions

Our contributions are split into two main parts. The first part of the thesis is focused on explaining human behavior and well-studied decision biases by viewing people as computationally-bounded. The second part of the thesis models players as polynomial-time TMs and develops tools to analyze such models, as well as study the implications of some of these models on fundamental questions in the intersection of game theory and computer science.

1.2.1 Human behavior as rational bounded agents

On the advantage of having high computational cost (Section 3)

We study a very basic question: does having high computational costs always hurt you? Or in other words, given an opportunity to reduce her costs, will a player always accept that? We first give simple examples where even if we improve a player's utility in every action profile, her payoff in equilibrium might be lower than in the equilibrium before the change. This is because keeping her high cost gives her a credible threat against the other players. We pro-

vide some conditions on games that are sufficient to ensure this does not occur, which basically correspond to having strict competition. We then show how this counter-intuitive phenomenon can explain real life phenomena such as free riding, and why this might cause people to give signals indicating that they are not as "smart" as they really are.

Modeling people as rational Finite automata in dynamic environments (Section 4)

We show that by modeling people as bounded finite automata, we can capture at a qualitative level human behavior observed in experiments. We consider a decision problem with incomplete information and a dynamically changing world, which can be viewed as an abstraction of many real-world settings. We provide a simple strategy for a finite automaton in this setting, and show that it does quite well, both through theoretical analysis and simulation. Thus, although simple, the strategy is a sensible strategy for a resource-bounded agent to use. Moreover, at a qualitative level, the strategy does exactly what people have been observed to do in experiments.

1.2.2 Models for computationally bounded agents

Computing a NE for repeated games played by polynomial-time TMs (Section 5)

We study the problem of computing an ϵ-Nash equilibrium in repeated games. Earlier work by Borgs et al. [8] suggests that this problem is intractable. We

show that if we make a slight change to their model—modeling the players as polynomial-time Turing machines that maintain state —and make some standard cryptographic hardness assumptions (the existence of public-key encryption), the problem can actually be solved in polynomial time. Our algorithm works not only for games with a finite number of players, but also for constant-degree graphical games.

As Nash equilibrium is a weak solution concept for extensive-form games, we additionally define and study an appropriate notion of a subgame-perfect equilibrium for computationally bounded players, and show how to efficiently find such an equilibrium in repeated games (again, making standard cryptographic hardness assumptions).

A model for extensive-form games played by polynomial-time TMs (Section 6)

We define a model of a *computational game*, which is a sequence of games that get larger in some appropriate sense, aimed at modeling computationally bounded players playing a fixed finite game. We define what it means for a computational game to represent a single finite underlying extensive-form game. Roughly speaking, we require all the games in the sequence to have essentially the same structure as the underlying game, except that two histories that are indistinguishable (i.e., in the same information set) in the underlying game may correspond to histories that are only computationally indistinguishable in the computational game.

We define a computational version of both Nash equilibrium and sequential

equilibrium for computational games, and show that every Nash (resp., sequential) equilibrium in the underlying game corresponds to a computational Nash (resp., sequential) equilibrium in the computational game.

One advantage of our approach is that if a cryptographic protocol represents an abstract game, then we can analyze its strategic behavior in the abstract game, and thus separate the cryptographic analysis of the protocol from the strategic analysis.

1.3 Bibliographic Notes

The material of this thesis is largely based on material from published or completed papers [62, 30, 31, 32, 33, 34]. The results of section 3 appeared in [62]. The results of section 4 appeared in [30, 31]. The results of section 5 appeared in [32, 33]. The results of section 6 appeared in [34].

CHAPTER 2

PRELIMINARIES

2.1 Normal-Form Games

We define a game G to be a triple $([c], A, \vec{u})$, where $[c] = \{1, \ldots, c\}$ is the set of players, A_i is the set of possible actions for player i, $A = A_1 \times \ldots \times A_c$ is the set of action profiles, and $\vec{u} : A \to \mathbb{R}^c$ is the utility function ($\vec{u}_i(\vec{a})$ is the utility of player i). A (mixed) *strategy* σ_i for player i is a probability distribution over A_i, that is, an element of $\Delta(A_i)$ (where, as usual, we denote by $\Delta(X)$ the set of probability distributions over the set X). We use the standard notation $\vec{x}_{-i}$ to denote vector $\vec{x}$ with its ith element removed, and $(x', \vec{x}_{-i})$ to denote $\vec{x}$ with its ith element replaced by x'.

Definition 2.1.1 *(Nash Equilibrium) $\sigma = (\sigma_1, ..., \sigma_c)$ is an ϵ-NE of G if, for all players $i \in [c]$ and all actions $a'_i \in A_i$, $E_{\sigma_{-i}}[u_i(a'_i, \vec{a}_{-i})] \leq E_{\sigma}[u_i(\vec{a})] + \epsilon$.*

A *correlated strategy* of a game G is an element $\sigma \in \Delta(A)$. It is a *correlated equilibrium* if, for all players i, they have no temptation to play a different action, given that the action profile was chosen according to σ. That is, for all players i for all $a_i \in A_i$ such that $\sigma_i(a_i) > 0$, $E_{\sigma|a_i} u_i(a_i, \vec{a}_{-i}) \geq E_{\sigma|a_i} u_i(a'_i, \vec{a}_{-i})$.

Player i's minimax value in a game G is the highest payoff i can guarantee himself if the other players are trying to push his payoff as low as possible. We call the strategy i plays in this case a minimax strategy for i; the strategy that the other players use is i's (correlated) punishment strategy. (Of course, there could be more than one minimax strategy or punishment strategy for player

i.) Note that a correlated punishment strategy can be computed using linear programming.

Definition 2.1.2 *Given a game $G = ([c], A, \vec{u})$, the strategies $\vec{\sigma}_{-i} \in \Delta(A_{-i})$ that minimize $\max_{\sigma' \in \Delta(A_i)} E_{(\sigma', \vec{\sigma}_{-i})}[u_i(\vec{a})]$ are the* punishment strategies *against player i in G. If $\vec{\sigma}_{-i}$ is a punishment strategy against player i, then $mm_i(G) = \max_{a \in A_i} E_{\vec{\sigma}_{-i}}[u_i(a, a_{-i})]$ is player i's* minimax value *in G.*

Part II

Human behavior as rational bounded agents

CHAPTER 3

I'D RATHER STAY STUPID: ON THE ADVANTAGE OF HAVING HIGH COMPUTATIONAL COST

3.1 Introduction

We are all familiar with situations where we feel that if only that other choice was a little cheaper, we could have done so much better, or if our computer was just a bit faster, we would have been in a much better situation when facing our competitors, or if the government would have just subsidized our research, we could have been in a much better position to compete. We next show that sometimes we were just wrong.

Our motivation for looking at this question comes from trying to understand if having bounded rationality is always bad for a player. More specifically, we consider whether decreasing the cost of computation would make an agent better off, and most of the examples we give are motivated by this question. However, our ideas can easily be generalized to any change in the utility of the players.

When players are bounded to use a fixed sized automaton, Ben-Porath [3] and Gilboa and Samet [19] showed, as our intuition expects, that a bounded player has disadvantages against a much stronger player who can use a much larger automaton. But Gilboa and Samet also showed a somewhat opposite phenomenon, which they called "the tyranny of the weak". They showed that the bounded player might actually gain from being bounded, relative to a situation where she was unbounded, because being bounded serves as a credible

"threat".

We explore this counter-intuitive scenario in the related framework, where, instead of being bounded, the players pay for the complexity of the strategy they use. Specifically, we look at the model of Ben-Sason, Kalai and Kalai [4], where, in addition to the game, each player has a cost associated with each action she is playing, regardless of the other players' strategies. We use a version of this framework to explore the added value for a player from having better complexity.

Our result are similar in spirits to results obtained for "value of information", the added value in expected utility that an agent gets from having information revealed to her. Blackwell [6, 7] showed that in a single-agent decision problem, the value of information is non-negative. In a multi-agent environment, the situation is more complicated, because we need to consider how the information that an agent possesses affects other agents' actions. Hirshleifer [37] and Kaimen, Tauman, and Zamir [44] showed that, in a multi-agent game, more information to a single player can result in an equilibrium in which her payoff is reduced. Neyman [55] showed that if the other players do not know about a player's new information, that player's payoff can not be reduced in equilibrium.

Similarly to the idea of value of information we compare games before and after a change in the utility functions of the players. As Neyman [55] pointed out, changing a game in such a way creates a totally new game, so by comparing the utility in equilibrium before and after the change, we actually compare two different games (for example, the information of the players also changes, since the game description is different and this is part of their information), so per-

haps it is not surprising that the results are not always what we might expect. Nevertheless, we also choose this approach since we feel that, although these are two different games, their most significant difference is the utility change and all other changes are caused by it.

We show that decreasing a player's computation cost (more generally, locally improving a player's utility) can lead her to a worse global outcome in equilibrium (When more than one equilibrium exists, we compare the equilibrium with the worst outcome for the player). The player actually loses some advantages she had from being weak. We also discuss some conditions under which this can not happen. These conditions correspond to a game with strict competition.

We then show how this unintuitive phenomenon, that at first might seem like a problematic aspect of the Nash equilibrium solution concept, can actually explain real life phenomena. We first show that this can explain free riding, where a group lets a weak member of it, that does not contribute for the group's effort, receive credit for the group's success. Moreover the weak player has no incentive to improve and become stronger. We then take an extra step and show that this advantage of weak players might cause people to give signals indicating, that they are "stupider" than they really are.

3.2 Example: I dare you to factor

Consider the following game G:

	factor	don't factor
factor	1,1	1,3
don't factor	3,1	-10,-10

This is an instance of the "chicken" game, where both players are presented with one large number to factor. A player who factors the number gets a reward of 1. If one player factors the number and the other does not, then the player who does not factor gets 3. However if neither player factors the number, they are both punished and need to pay 10. This game has two pure-strategy equilibria in which one player factors and the other doesn't factor, and one mixed-strategy equilibrium where they both factor with probability $\frac{11}{13}$ and get an expected reward of 1.

Now consider the following : The year is 2040, Player 1 has a powerful state-of-the-art classical computer, while player 2 has the newest "Ox" quantum computer, capable of factoring very large numbers efficiently. Both players have a complexity cost associated with every action they take that is represented as a complexity function. Player 1 has a complexity function c_1, where not factoring cost nothing, and factoring is not possible, so its complexity is ∞. Player 2's complexity function is 0 for both actions. A player's utility is simply the reward of the player minus her complexity cost. This game has only one equilibrium: player 1 does not factor, while player 2 factors. The utility vector in this equilibrium is $(3, 1)$.

Now what happens if we change player 1's complexity function by giving her an "Ox" computer? Her utilities have obviously improved everywhere, but does this help her or hurt her? The new game we get is identical to the original game without complexity costs, and so has two more equilibria . In both new

equilibria player 1's utility is only 1 instead of 3. The third equilibrium is identical to the equilibrium with the old costs. This change made things worse for player 1, since in the worst case her utility with the new complexity function is lower than the utility with the old complexity function. What actually happens here is that when player 1 has only a classical computer, she has a credible "threat": she is not going to factor no matter what player 2 does (This is similar to removing the stirring wheel from the car in the traditional story of the chicken game). Thus player 2 must factor. When they both have the "Ox" computer, that threat is gone and player 2 is not going to agree to always factor. If offered a free "Ox" computer, player 1 will actually refuse to get it.

The next example shows that the player can do strictly worse in all equilibria by this kind of change to the utilities, not only in the worst case equilibrium. Consider the following game:

	a_2	b_2
a_1	2,1	-2,2
b_1	3,1	-1,-1

In this game, player 1 has a dominant strategy b_1, which leads to only one equilibrium, (b_1,a_2), with utilities $(3, 1)$. Now if player 1 gets a subsidy of 2 when playing a_1, we get a different equilibrium, (a_1,b_2), with utilities of $(0, 2)$. Note that even the social welfare is worse in this scenario. This change happens because player's 1 dominant strategy changed from b_1 to a_1. When player 1 plays b_1, player 2 prefers to play a_2. The change in utility for player 1 changes the dynamics between the two players, which makes player 2 also change her actions, and leads to a new equilibrium. Getting the subsidy, which improved player 1 utilities locally, leads to an equilibrium where her utility is lower. What

actually happens is that when player 1 gets no subsidy for playing a_1, player 2 knows she can't make player 1 play anything that is not b_1, so she has no choice but to play a_2. When she does get the subsidy for a_1, player 2 knows player 1 will play a_1 always so she can play b_2. Player 1 can't threaten player 2 with playing b_1 any more.

These two examples show that changing a player's utilities so that she is better off in any strategy profile (improving her complexity function for every action) might result in an equilibrium in which the player's utility is lower than with her old utilities. This happens because having a bad utility for some profiles gives a player a threat against the other players. This threat is lost when her utility gets better, and some actions that were once unacceptable by her might now be a best response for her to the other players' choice of actions. This is used by the other players to change the equilibrium of the game and create a final result where the player might have lower utility. So, although the player is better off for any strategy profile chosen, the strategy profile that is an equilibrium in the new game is worse for her.

This section showed that, in general games, increasing a player's utility locally (or reducing her complexity cost) can result in an equilibrium where her utility is lower. In a sense, we showed that in some games players actually prefer to be bounded or weak. The next section considers at the special case of constant-sum games.

3.3 Constant-sum games

Constant-sum games have some unique characteristics. In particular they are totally competitive. No one can gain from cooperation. Our intuition is that in these kind of games, a player can not get hurt by improving her utility function. In this section, we show that this intuition is correct and what kind of changes can we make to such games and still have the same effect.

Constant-sum games have the very nice property that by the minimax theorem we know exactly how the players play at equilibrium. In particular, we know that in equilibrium each player plays her defense strategy (her maxmin strategy) which means she plays the strategy that maximizes her minimum payoff - the strategy that gives her the maximum payoff against any strategy the other player plays. We use this fact to show the following theorem.

Theorem 3.3.1 *Let G be a 2-player constant-sum game with utility functions u. Let G' be a game with the same action space as G but with utility functions u' such that for all $\overrightarrow{a}, u_i(\overrightarrow{a}) \leq u'_i(\overrightarrow{a})$, and u'_{-i} changed arbitrary. In equilibrium, player i's utility in G' can't be lower than in G.*

Proof: Without loss of generality, assume that $i = 1$. Now lets look at any equilibrium in G'. If player 2 plays the same strategy she plays in the equilibrium in G, then if player 1 plays the same strategy she plays in G, we know that her utility improved by definition. If she plays another strategy then by the definition of equilibrium she gets at least as much as with her strategy in G, otherwise she would want to switch. So in this case player 1's utility can't be lower.

If player 2 plays a different strategy than in G, then we know that if player

1 plays her strategy in G, she gets at least the same payoff. That is because we know that G is constant-sum, so player 2 minimizes player 1 payoff with that action in G, so if she now changes action, player 1 could only improve. Using the same argument as in the previous case, we know that if player 1 plays another strategy, then she must get more than in G. □

This shows that when starting from a constant-sum game, any change to the game that improves one player's utility for every action profile can't hurt her, no matter what changes are done to the other player. The next theorem shows that even games that are not exactly constant-sum but are close to them, have the same characteristics.

The games we consider are games with utility of the form $u_i(\overrightarrow{a}) = u_i^u(\overrightarrow{a}) - \xi_i(a_i)$, where u_i^u is the utility player i gets if there was no cost involved, and we assume the sum of u_i^u over all players is constant for any action profile. With every action a, player i has an associated cost (or subsidy) $\xi_i(a)$, and the player does not gain from other players' costs. These games are the same as the games studied by Ben-Sason, Kalai and Kalai [4]. We show that if only one of the players has a cost for her actions (or the other player has a constant cost, which is just a constant-sum game with a different constant) then that player can not lose from changes that improve her utility.

Theorem 3.3.2 *Let G be a 2-player constant-sum game, with utility functions $\overrightarrow{u^u}$. In a game G^ξ in which the utility functions $\overrightarrow{u}$ are of the form $u_i(\overrightarrow{a}) = u_i^u(\overrightarrow{a}) - \xi_i(a_i)$, and for one of the players ξ_i is constant, the other player can not lose in equilibrium from a local improvement of its cost function.*

Proof: First assume without loss of generality that player 2's cost func-

tion is constant with a value of c, and player 1's cost improved. We use the same idea as Ben-Sason, Kalai and Kalai [4]. Given a game G^ξ, we build a game H, that differs from G^ξ only in the utilities: we define $u_i^h(\overrightarrow{a}) = u_i(\overrightarrow{a}) + \xi_{-i}(a_{-i}) = u_i^u(\overrightarrow{a}) - \xi_i(a_i) + \xi_{-i}(a_{-i})$ for $i = 1, 2$. It easy to verify that any advantage a player can get from switching strategies in H, she can get from switching strategies in G^ξ. This means that any equilibrium in H is an equilibrium in G^ξ, and vice versa. Let σ_1, σ_2 be the strategies at an equilibrium that is worst for player 1 in H, and let $P_{\sigma_i}(a)$ be the probability of playing action a when using strategy σ_i. Then:

$$
\begin{aligned}
u_1(\sigma_1, \sigma_2) &= \sum_{a,b} p_{\sigma_1}(a) p_{\sigma_2}(b) u_1(a, b) \\
u_1^h(\sigma_1, \sigma_2) &= \sum_{a,b} p_{\sigma_1}(a) p_{\sigma_2}(b) u_1^h(a, b) \\
&= \sum_{a,b} p_{\sigma_1}(a) p_{\sigma_2}(b) u_1(a, b) + \sum_{a,b} p_{\sigma_1}(a) p_{\sigma_2}(b) \xi_2(b) \\
&= \sum_{a,b} p_{\sigma_1}(a) p_{\sigma_2}(b) u_1(a, b) + \xi \text{ (ξ_2 is constant)} \\
&= u_1(\sigma_1, \sigma_2) + \xi
\end{aligned}
$$

This shows that player 1's utility in any equilibrium in H is her utility in the same equilibrium in G^ξ plus some constant. Moreover when comparing any two equilibria in H, the difference in the utility of player 1 in them is exactly the difference in her utility in G^ξ. This means that σ_1, σ_2 is also the worst equilibrium for player 1 in G^ξ.

H is a game of the type described in Theorem 1. So, by that theorem, if we change player 1's cost function ξ_1 to a cost function ξ_1', where she pays no more than with ξ_1 for her actions, player 1 gets at least the same utility in the

worst case. We call the games created by changing ξ_1 to ξ_1' H' and $G^{\xi'}$. By the same argument as before, the new worst-case equilibrium for player 1 in H' is also the worst-case equilibrium for player 1 in $G^{\xi'}$, and the difference between the utility of player 1 in the worst-case equilibrium for player 1 of H and H' is exactly equal to the differences between player 1's utility in the worst-case equilibrium for player 1 of G^{ξ} and $G^{\xi'}$. This means that in $G^{\xi'}$ she is also at least as well off as she was in G^{ξ}. □

The next example shows that if both players have non-constant costs this property fails to hold. Consider the following game:

u^u	a_2	b_2
a_1	6,0	2,4
b_1	4,2	1,5

where the costs are 0 for all actions for all players. The game described has only one equilibrium: (a_1, b_2) with utility profile $(2, 4)$. Now consider the situation where $\xi_1(a_1) = -1.5$, $\xi_1(b_1) = 0$, $\xi_2(a_2) = 0$, $\xi_2(b_2) = -3.5$. This game has only one equilibrium: both players play each action with probability 0.5. The expected utility is (2.5,1). By changing only ξ_1 to ξ_1', which is 0 for both actions, which obviously locally improve the player's utility, the equilibrium is changed back to (a_1, b_2) with utilities $(2, 0.5)$, and thus the player is globally worse than before.

This section shows how in constant-sum games, a player can always gain from a local improvement in her utility, and that it is also true in games which are close to constant sum games (a change to only one player makes them constant-sum again). There are of course other changes that can not harm a

player, even in games which are not constant sum. For example, reducing the cost of all actions in the same amount (which is just like giving free money no matter what the player does), or improving the cost of an already dominant strategy by more than that of other strategies.

3.4 Explaining human behavior as results of this phenomenon

3.4.1 I would rather stay stupid

The phenomenon of free riding is well observed and studied. One flavor of it occurs when a part of a group gets credit for the work of others without contributing anything. We argue that this can be explained by the weak player's advantage we described, and moreover that it also explains why there is a negative incentive for weak players to improve. We illustrate this by an example that shows how a weak player gets credit without any contribution, and why the rest of the group might agree to it.

Consider a scenario where two students have to work on an assignment together, but they can't meet and exchange work before its deadline. The assignment has 10 questions. In order to solve each question, you must first have the answer to the previous question. Both students gain one point for every question any one of them solves; if they both solve the same question they still get only 1 each. This means that their utility can be written as $u = max(x_1, x_2)$ where x_i is the number of questions student i solves. The first student has a complexity cost of 0.1 for every question she solves, while the second student

has a complexity cost of 0.1 for the first 7 question, and a cost of 1.1 for the rest. This game has only one equilibrium: the first student does all 10 questions, while the second student free rides (does nothing) and gets the credit. This is because for every question the second student will solve after the 7th she will get -0.1 utility, so she will not do more then 7 questions, while the first student prefers solving 10 question by herself to doing nothing and having the second student solve only 7 questions. The utility for the second student in this game is 10.

Now what would have happened if the second student were "smarter", and had the same complexity cost as the first student? In this situation, the game would have one more pure strategy equilibrium (it also has a mixed strategy equilibrium), in which the second student solves all 10 questions and the first student does nothing. In this equilibrium, the second student's utility is 9, which is lower than before. This shows that the second student has no incentive to try to get "smarter". By free riding, she ensures herself the highest utility possible in this game, while the other student has no choice but to do all the work. This happens because the second student has a credible threat: no matter what the first student does, she won't solve more than 7 questions. This shows that the second student has no incentive to improve. If she gets smarter she actually loses the threat and gets a lower payoff.

3.4.2 I am better than I seem

In this section, we explore why people sometimes pretend to be weak, when they are really not, which is also a well observed phenomenon. Intuitively, we

show that the reason for this is that acting weak, lowers the expectation from the players, and allows them to invest less effort. To do that we do not look at the added value a player gets from better utility, but instead look at a scenario where a player would rather behave as if she had lower utility, since in equilibrium it gets her a higher payoff.

We use the spirit of Spence's [65] signaling model, which is traditionally used to show how players signal how good they are to the other players, to instead show that players might sometimes want to do the opposite, and signal that they are even worse then they really are. Spence shows how education can be seen as a signal in the hiring market, which helps employers decides the wages to offer job candidates. He defines an information-feedback cycle, consisting of the employer's beliefs, the offered wages as a function of the signals, the signal chosen by applicants, and the final observation of the hired employees by the employers (which feeds back into their beliefs). He defines an equilibrium in this model as a situation where the beliefs of the employer are self confirming, that is, do not change as a result of the final observation of the employers that were hired based on the previous beliefs.

We use a variant of this model to show that people might even try to seem stupider than they really are (or more generally signal that their utility\complexity functions are weaker) to get the power of a credible threat. Consider an educational institution that wants to divide its students into two classes, regular and honors. For every student that is placed in the honors class and is able to pass it, the school gets a utility of 1, but if the student is placed in the honors class and fails, the school get a utility of -1. For every student that is placed in the regular class, the school gets a utility of 0. A student has two

options: she can either relax and easily pass the regular class but fail the honors class, or she can work hard, in which case she passes both classes. If she passes a class she gets utility of 100, and if she fails she gets utility of 0.

There are three types of students. A slow student, who has a cost of 100 for working hard, a moderate student, who has a cost of 7, and a fast student, who has a cost of 3. The cost of relaxing is 0 for all students. The school would like to place the slow students in the regular class and the moderate and fast students in the honors class, but it cannot tell which students fall into each category.

To help it with the process, the school decides to do a preliminary placement test for students. The test has 10 questions, and the school decides that a student who solves 7 or more questions will be placed in the honors class. To motivate the students to perform well, the school offers them the option of skipping one hour of class without being punished for every question they solve. Skipping an hour has a utility of 1. The three types of students have different costs for this test. All students have a cost of 0 for the first six questions. For every question after that, the slow student has a cost of 1.1, the moderate student has a cost of 0.5, and the fast student has a cost of 0.2. The school is unaware of these exact costs (as in the Spence model, where the employer is unaware of the exact education costs of the different groups), but designs the test knowing that both the fast and moderate students have a positive incentive to answer all the questions, while the slow student will not answer more than six.

It is easy to see that in order to maximize their utility, the slow and moderate students will answer only six questions (giving them utility 106), and the fast students will answer all ten questions (giving them utility 106.2). Doing badly in the exam acts as a signal of being slower (and is negatively correlated with

the utility, as required by the Spence model). Since the only way for the school to figure out if it did the right thing is to see if someone failed the honors class (this is slightly different from Spence's original model, where the employer gets complete feedback), and no students in the honors class will fail, the school's beliefs are self-confirming, so this gives an equilibrium.

As in our previous example, the slow students have no motivation to become moderate, thus changing from the cost of the slow student to that of the moderate student has value of of 0. The value for both the moderate and the slow student of getting the complexity of the fast player is positive in this example.

This example shows that people will sometimes prefer to be considered less smart then they really are, in order to take advantage of the threat of having higher complexity.

CHAPTER 4

I'M DOING AS WELL AS I CAN: MODELING PEOPLE AS RATIONAL FINITE AUTOMATA

4.1 Introduction

Our goal in this section is to better understand how people make decisions in dynamic situations with uncertainty. There are many examples known where people do not seem to be choosing strategies that maximize expected utility. Various approaches have been proposed to account for the behavior of people, of which perhaps the best known is Kahnemann and Tversky's [42] *prospect theory*. As we discussed before, one explanation for this inconsistency between expected utility theory and real-life behavior has been that agents are *boundedly rational*—they are rational, but computationally bounded. One of the most commonly-used model of computationally bounded agents has been finite automata.

Wilson [69] considers a decision problem where an agent needs to make a single decision, whose payoff depends on the state of nature (which does not change over time). Nature is in one of two possible states, G (good) and B (bad). The agent gets signals, which are correlated with the true state, until the game ends, which happens at each step with probability $\eta > 0$. At this point, the agent must make a decision. Wilson characterizes an n-state optimal finite automaton for making a decision in this setting, under the assumption that η is small (so that the agent gets information for many rounds). She shows that an optimal n-state automaton ignores all but two signals (the "best" signal for each of nature's states); the automaton's states can be laid out "linearly", as states

0, ..., $n-1$, and the automaton moves left (with some probability) only if it gets a strong signal for state G, and moves right (with some probability) only if it gets a strong signal for state B. Thus, roughly speaking, the lower the current state of the automaton, the more likely from the automaton's viewpoint that nature's state is G. (Very similar results were proved earlier by Hellman and Cover [36].) Wilson argues that these results can be used to explain observed biases in information processing, such as *belief polarization* (two people with different prior beliefs, hearing the same evidence, can end up with diametrically opposed conclusions) and the *fist-impression bias* (people tend to put more weight on evidence they hear early on). Thus, some observed human behavior can be explained by viewing people as resource-bounded, but rationally making the best use of their resources (in this case, the limited number of states).

Wilson's model assumes that nature is static. But in many important problems, ranging from investing in the stock market to deciding which route to take when driving to work, the world is dynamic. Moreover, people do not make decisions just once, but must make them often. For example, when investing in stock markets, people get signals about the market, and need to decide after each signal whether to invest more money, take out money that they have already invested, or to stick with their current position.

Here we consider a model that is intended to capture the most significant features of a dynamic situation. As in Wilson's model, we allow nature to be in one of a number of different states (for simplicity, like Wilson, for most parts we assume that nature is in one of only two states), and assume that the agent gets signals correlated with nature's state. But now we allow nature's state to change, although we assume that the probability of a change is low. (Without

this assumption, the signals are not of great interest.)

Our choice of model is in part motivated by recent work by psychologists and economists on how people behave in such scenarios, particularly that of Erev, Ert, and Roth [15], who describe contests that attempt to test various models of human decision making under uncertain conditions. In their scenarios, people were given a choice between making a safe move (that had a guaranteed constant payoff) and a "risky" move (which had a payoff that changed according to an unobserved action of the other players). Since their goal was that of finding models that predicted human behavior well, Erev et al. [15] considered a sequence of settings, and challenged others to present models that would predict behavior in these settings.

They also introduced a number of models themselves, and determined the performance of these models in their settings. One of those models is *I-Saw* (inertia, sampling, and weighting) [15]; it performed best among their models, with a correlation of 0.9 between the model's prediction and the actual observed results for most variables. I-Saw assumes that agents have three types of response mode: *exploration*, *exploitation*, and *inertia*. An I-Saw agent proceeds as follows. The agent tosses a coin. If it lands heads, the agent plays the action other than the one he played in the previous step (*exploration*); if it lands tails, he continues to do what he did in the previous step (*inertia*), unless the signal received in the previous round crosses a probabilistic "surprise" trigger (the lower the probability of the signal to be observed in the current state, the more likely the trigger is to be crossed); if the surprise trigger is crossed, then the agent plays the action with the best estimated subjective value, based on some sampling of the observations seen so far (*exploitation*).

The winner of the contest was a refinement of I-Saw called *BI-Saw* (bounded memory, inertia, sampling and weighting) model, suggested by Chen et al. [9]. The major refinement involved adding a bounded memory assumption, whose main effect is a greater reliance on a small sample of past observations in the exploitation mode. The *BI-Saw* model had a normalized mean square deviation smaller than 1.4 for estimating the entry rate of the players, and smaller than 1 for estimating the actual payoff they get, which was better than the results of the *I-Saw* model.

I-Saw and BI-Saw seem quite *ad hoc*. We show that they can be viewed as the outcomes of play of a resource-bounded agent modeled as a finite automaton.[1] Specifically, we consider a setting where an agent must make a decision every round about playing safe (and getting a guaranteed payoff) or playing a risky move, whose payoff depends on the state of nature. We describe a simple strategy for this agent, and show both theoretically and by simulation that it does very well in practice. While it may not be the optimal strategy if the agent is restricted to n states, we show that as n goes to infinity and the probability of nature changing state goes to 0, the expected payoff of this strategy converges to the best expected payoff that the player could get even if he knew the state of nature at all times. Interestingly, this strategy exhibits precisely the features of (I-Saw and) BI-Saw at a qualitative level. Thus, we believe that (B)I-Saw can be best understood as the outcome of a resource-bounded agent playing quite rationally.

[1] Although the scenarios in [15] are games rather than decision problems, as observed in [16], learning in a decision problem should work essentially the same way as learning in games, so, for simplicity, we consider the former setting. The winner of a similar contest for single agent decision problems (see Erev et al. [17]) was won by a predecessor of the I-Saw model with very similar behavior modes. In our setting, the agent's actions do not influence nature, which is similar to assumptions usually made in large economic markets, where a single agent cannot influence the market.

4.2 The Model

We assume that nature is in one of two states G ("good") and B ("bad"); there is a probability π of transition between them in each round. (In Section 4.4, we show that allowing nature to have more states does not affect the results at a qualitative level; similarly, while we could allow different transition probabilities from G to B and from B to G, this would not have an impact on our results.)

The agent has two possible actions S (safe) and R (risky). If he plays S, he gets a payoff of 0; if he plays R he gets a payoff $x_G > 0$ when nature's state is G, and a payoff $x_B < 0$ when nature's state is B. The agent does not learn his payoff, and instead gets one of k signals. Signal i has probability p_i^G of appearing when the state of nature is G, and probability p_i^B of appearing when the state is B. We assume that the agent gets exactly one signal at each time step, so that $\sum_{i=1}^k p_i^G = \sum_{i=1}^k p_i^B = 1$. This signaling mechanism is similar to that considered by Cover and Hellman [36]. However, we assume that the agent gets a signal only if he plays the risky action R; he does not get a signal if he plays the safe action S. We denote this setting $S[p_1^G, p_1^B, \ldots, p_k^G, p_k^B, x_G, x_B]$. We say that a setting is *nontrivial* if there exists some signal i such that $p_i^B \neq p_i^G$. If a setting is trivial, then no signal enables the agent to distinguish whether nature is in state G or B; the agent does not learn anything from the signals. (Note that we have deliberately omitted nature's transition probability π from the description of the setting. That is because, in our technical results, we want to manipulate π while keeping everything else fixed.) One quick observation is that a trivial strategy that plays one of S or R all the time gets a payoff of 0 in this model, as does the strategy that chooses randomly between S and R. We are interested in finding

a simple strategy that gets appreciably more than 0.[2]

As suggested by the BI-Saw model, we assume that agents have bounded memory. We model this by assuming agents which are restricted to using a finite automaton, with deterministic actions and probabilistic transitions, and a fixed number n of internal states. The agent's goal is to maximize his expected average payoff.

We focus on one particular family of strategies (automata) for the agent. We denote a typical member of this family $A[n, p_{exp}, Pos, Neg, r_u, r_d]$. The automaton $A[n, p_{exp}, Pos, Neg, r_u, r_d]$ has $n + 1$ states, denoted $0, \ldots, n$. State 0 is dedicated to playing S. In all other states R is played. The k signals are partitioned into three sets, Pos (for "positive"), Neg (for "negative"), and I (for "ignore" or "indifferent"), with Pos and Neg nonempty. Intuitively, the signals in Pos make it likely that nature's state is G, and the signals in Neg make it likely that the state of nature is B. The agent chooses to ignore the signals in I; they are viewed as not being sufficiently informative as to the true state of nature. (Note that I is determined by Pos and Neg.)

In each round while in state 0, the agent moves to state 1 with probability p_{exp}. In a state $i > 0$, if the agent receives a signal in Pos, the agent moves to $i + 1$ with probability r_u (unless he is already in state n, in which case he stays in state n if he receives a signal in Pos); thus, we can think of r_u as the probability the the agent moves up if he gets a positive signal. If the agent receives a signal in Neg, the agent moves to state $i - 1$ with probability r_d (so

[2] This model can be viewed as an instance of a "one-armed restless bandit" [68] that does not have perfect information about the state of the project. This kind of decision problem was also tested [5], and the model that performed best can be viewed as a predecessor of the I-Saw model. A similar model was also studied in animal psychology as a model of animal food gathering [51].

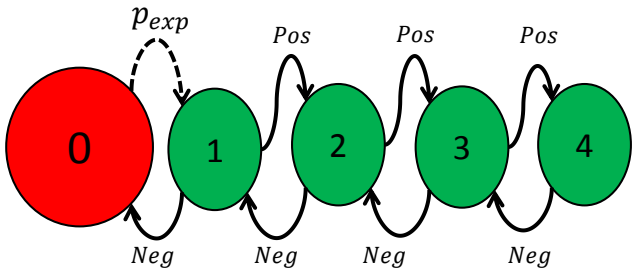

Figure 4.1: Example of automaton $A[4, p_{exp}, Pos, Neg, 1, 1]$

r_d is the probability of moving down if he gets a signal in Neg); if he receives a signal in I, the agent does not changes states. Clearly, this automaton is easy for a human to implement (at least, if it does not have too many states). See Figure 4.1 for an example of such an automaton.

Note that this automaton incorporates all three behavior modes described by the I-Saw model. When the automaton is in state 0, the agent *explores* with constant probability by moving to state 1. In state $i > 0$, the agent continues to do what he did before (in particular, he stays in state i) unless he gets a "meaningful" signal (one in Neg or Pos), and even then he reacts only with some probability, so we have *inertia*-like behavior. If he does react, he *exploits* the information he has, which is carried by his current state; that is, he performs the action most appropriate according to his state, which is R. The state can be viewed as representing a sample of the last few signals (each state represents remembering seeing one more "good" signal), as in the BI-Saw model.

4.3 Theoretical analysis

In this section, we do a theoretical analysis of the expected payoff of the automaton $A[n, p_{exp}, Pos, Neg, r_u, r_d]$. This will tell us how to optimally choose the parameters Pos, Neg, r_u, and r_d. Observe that the most any agent can hope to get is $\frac{x_G}{2}$. Even if the agent had an oracle that told him exactly what nature's state would be at every round, if he performs optimally, he can get only x_G in the rounds when nature is in state G, and 0 when it is in state B. In expectation, nature is in state G only half the time, so the optimal expected payoff is $x_G/2$. One of our results shows that, somewhat surprisingly, as n gets large, if π goes to 0 sufficiently quickly, then the agent can achieve arbitrarily close to the theoretical optimum using an automaton of the form $A[n, p_{exp}, Pos, Neg, r_u, r_d]$, even without the benefit of an oracle, by choosing the parameters appropriately. More precisely, we have the following theorem.

Theorem 4.3.1 *Let* Π *and* P_{exp} *be functions from* $\mathbb{N}$ *to* $(0, 1]$ *such that* $\lim_{n\to\infty} n\Pi(n) = \lim_{n\to\infty} \Pi(n) = \lim_{n\to\infty} \Pi(n)/P_{exp}(n) = 0$. *Then for all settings* $S[p_1^G, p_1^B, \ldots, p_k^G, p_k^B, x_G, x_B]$, *there exists a partition* Pos, Neg, I *of the signals, and constants* r_d *and* r_u *such that* $\lim_{n\to\infty} E_{\Pi(n)}[A[n, P_{exp}(n), Pos, Neg, r_u, r_d]] = \frac{x_G}{2}$.

Note that in Theorem 4.3.1, $\Pi(n)$ goes to 0 as n goes to infinity. This requirement is necessary, as the next result shows; for fixed π, we can't get too close to the optimal no matter what automaton we use (indeed, the argument applies even if the agent uses a Turing machine instead of a finite automaton).

Theorem 4.3.2 *For all fixed* $0 < \pi \leq 0.5$ *and all automata* A, *we have* $E_\pi[A] \leq x_G/2 + \pi x_B/2$.

Proof: Suppose that the automaton had an oracle that, at each time t, correctly told it the state of nature at the previous round. Clearly the best the automaton could do is to play S if the state of nature was B and play R if the state of nature was G. Thus, the automaton would play R half the time and G half the time. But with probability π the state of nature will switch from G to B, so the playoff will be x_B rather than x_G. Thus, the payoff that it gets with this oracle is $x_G/2 + x_B\pi/2$. (Recall that $x_B < 0$.) We can think of the signals as being imperfect oracles. The automaton will do even worse with the signals than it will be oracle. □

The theorem focuses on small values of π, since this is our range of interest. We can prove a result in a similar spirit even if $0.5 < \pi < 1$.

The key technical result, from which Theorem 4.3.1 follows easily, gives us a very good estimate of the payoff of the automaton $A[n, p_{exp}, Pos, Neg, r_u, r_d]$, for all choices of the parameters. We state the estimate in terms of n, π, p_{exp} and four auxiliary quantities, ρ_u^G, ρ_u^B, ρ_d^G, and ρ_d^B, Intuitively, ρ_u^N is the probability of the automaton changing states from i to $i+1$ (going "up") when nature is in state N and $i \geq 1$, and ρ_d^N is the probability of the automaton changing states from i to $i-1$ (going "down") given that nature is in state N. Thus, $\rho_u^N = (\sum_{i \in Pos} p_i^N) r_u$ and $\rho_d^N = (\sum_{i \in Neg} p_i^N) r_d$. We define $\sigma_N = \rho_u^N / \rho_d^N$. Recall that when the automaton is in state 0, it does not get any signals; rather, it explores (moves to state 1) with probability p_{exp}.

Proposition 4.3.3

$$
\begin{aligned}
&E_\pi[A[n, p_{exp}, Pos, Neg, r_u, r_d]] \geq \\
&\tfrac{x_G}{2}\left(1 - \tfrac{(\rho_u^G - \rho_d^G) + \pi(\sum_{i=1}^n (\sigma_G)^i - n)}{(\rho_u^G - \rho_d^G) + p_{exp}((\sigma_G)^n - 1)}\right) + \\
&\tfrac{x_B}{2}\left(1 - \tfrac{(\rho_u^B - \rho_d^B) - \pi(\sum_{i=1}^n (\sigma_B)^i - n)}{(\rho_u^B - \rho_d^B) + p_{exp}((\sigma_B)^n - 1)}\right).
\end{aligned}
\tag{4.1}
$$

We sketch a proof of Proposition 4.3.3 in the next section. Although the expression in (4.1) looks rather complicated, it gives us just the information we need, both to prove Theorem 4.3.1 and to define an automaton that does well even when n is finite (and small).

Proof of Theorem 4.3.1: We want to choose Pos, Neg, r_u, and r_d so that $\rho_u^G > \rho_d^G$—the agent is more likely to go up than down when nature is in state G (so that he avoids going into state 0 and getting no reward) and $\rho_d^B > \rho_u^B$—the agent is more likely to go down than up when nature is in state B (so that he quickly gets into state 0, avoiding the payoff of -1). Suppose that we can do this. If $\rho_u^G > \rho_d^G$, then $\sigma_G > 1$, so the first term in the expression for the lower bound of $E_{\Pi(n)}[A[n, P_{exp}(n), Pos, Neg, r_u, r_d]]$ given by (4.1) tends to $\frac{x_G}{2}(1 - \frac{\Pi(n)}{P_{exp}(n)})$ as $n \to \infty$. Since we have assumed that $\lim_{n\to\infty} \Pi(n)/P_{exp}(n) = 0$, the first term goes to $x_G/2$. If $\rho_u^B < \rho_d^B$, then $\sigma_B < 1$, so the second term goes to

$$\frac{x_B}{2}\left(1 - \frac{(\rho_u^B - \rho_d^B) + \Pi(n)\frac{\sigma_B}{1-\sigma_B} + n\Pi(n)}{(\rho_u^B - \rho_d^B) - P_{exp}(n)}\right).$$

Since we have assumed that $\lim_{n\to\infty} n\Pi(n) = \lim_{n\to\infty} P_{exp}(n) = 0$, the second term goes to 0.

Now we show that we can choose Pos, Neg, r_B, and r_G so that $\rho_u^G > \rho_d^G$ and $\rho_d^B > \rho_u^B$. By assumption, there exists some signal i such that $p_i^G \neq p_i^B$. Since $\sum_{i=1}^k p_i^G = \sum_{i=1}^k p_i^B (= 1)$, it must be the case that there exists some signal i such that $p_i^G > p_i^B$. Let $Pos = \{i\}$. If there exists a signal j such that $p_j^G < p_i^B$ and $p_j^B > p_i^B$, then let $Neg = \{j\}$ and $r_u = r_d = 1$. Otherwise, let $Neg = \{1 \dots k\} \setminus \{i\}$, $r_u = 1$, and let r_d be any value such that $\frac{p_i^G}{1-p_i^B} < r_d < \frac{p_i^G}{1-p_i^G}$. It is easy to check that, with these choices, we have $\sigma_G > 1$ and $\sigma_B < 1$. This completes the proof

of Theorem 4.3.1. □

As we said, Proposition 4.3.3 gives us more than the means to prove Theorem 4.3.1. It also tells us what choices to make to get good behavior of n is finite. In Section 4.4, we discuss what these choices should be.

Proving Proposition 4.3.3

Once we are given π and a setting $S[p_1^G, p_1^B, \ldots, p_k^G, p_k^B, x_G, x_B]$, an automaton $A[n, p_{exp}, Pos, Neg, r_u, r_d]$ determines a Markov chain, with states of $(0, G), \ldots, (n, G), (0, B), \ldots, (n, B)$, where the Markov chain is in state (i, N) if nature is in state N and the automaton is in state i. The probability of transition is completely determined by π, the parameters of the automaton, and the setting.

Let $q_i^N(s, t)$ be the probability of the Markov chain being in state (i, N) at time t when started in state s. We are interested in $\lim_{t\to\infty} q_i^N(s, t)$. In general, this limiting probability may not exist and, even when it does, it may depend on the state the Markov chain starts in. However, there are well known sufficient conditions under which the limit exists, and is independent of the initial state s. A Markov chain is said to be *irreducible* if every state is reachable from every other state; it is *aperiodic* if, for every state s, there exist two cycles from s to itself such that the gcd of their lengths is 1. The limiting probability exists and is independent of the start state in every irreducible aperiodic Markov chain over a finite state space [58, Corollary to Proposition 2.13.5]. Our Markov chain is obviously irreducible; in fact, there is a path from every state in it to every other state. It is also aperiodic. To see this, note that if $0 < i \leq n$, there is a cycle of

length $2i$ that can be obtained by starting at (i, N), going to $(0, N)$ (by observing signals in Neg and nature not changing state) and going back up to (i, N). At $(0, N)$, there is a cycle of length 1. Thus, we can get a cycle of length $2i + 1$ starting at (i, N). Since we can go from $(0, B)$ to $(0, G)$ and back, there is also a cycle of length 2 from every state $(0, N)$. Since a limiting probability exists, we can write q_i^N, ignoring the arguments s and t.

We are particularly interested in the q_0^B and q_0^G, because these quantities completely determine the agent's expected payoff. As we have observed before, since the probability of transition from B to G is the same as the probability transition from G to B, nature is equally likely to be in state B and G. Thus, $\sum_{i=0}^{n} q_i^B = \sum_{i=0}^{n} q_i^G = 1/2$. Now the agent gets a payoff of x_G when he is in state $i > 0$ and nature is in state G; he gets a payoff of x_B when he is in state $i > 0$ and nature is in state B. Thus, his expected payoff is $x_G(1/2 - q_0^G) + x_B(1/2 - q_0^B)$.

It remains to compute q_0^B and q_0^G. To do this, we need to consider q_i^N for all values of i. We can write equations that characterize these probabilities. Let $\bar{N}$ be the state of nature other than N (so $\bar{B} = G$ and $\bar{G} = B$). Note that for a time t after (i, N) has reached its limiting probability, then the probability of state (i, N) has to be the same at time t and time $t + 1$. If $i > 0$, the probability of the system being in state (i, N) at time $t+1$ is the sum of the probability of (a) being in state $(i + 1, N)$ (or (n, N) if $i = n$), getting a signal in Neg and reacting to it, and nature not changing state, (b) being in state $(i-1, N)$, getting a signal in Pos and reacting to it (or, if $i = 1$, the system was in state $(i, 0)$ and the agent decided to explore), and nature did not change state, (c) being in state (i, N), getting a signal in I, and nature not changing state, (d) three further terms like (a)–(c) where the system state is $(j, \bar{N})$ at time t, for $j \in \{i - 1, i, i + 1\}$ and nature's

state changes. There are similar equations for the state $(0, N)$, but now there are only four possibilities: (a) the system was in state $(1, N)$ at time t, the agent observed a signal in Neg and reacted to it, and nature's state didn't change, (b) the system was in state $(0, N)$ and the agent's state didn't change, and (c) two other analogous equations where nature's state changes from $\bar{N}$ to N.

These considerations give us the following equations:

$$\begin{aligned}
q_0^N = \;& (1-\pi)((1-p_{exp})q_0^N + \rho_d^N q_1^N) \\
& +\pi((1-p_{exp})q_0^{\bar{N}} + \rho_d^{\bar{N}} q_1^{\bar{N}}) \\
q_1^N = \;& (1-\pi)((1-\rho_d^N-\rho_u^N)q_1^N + \rho_d^N q_2^N + p_{exp} q_0^N) \\
& +\pi((1-\rho_d^{\bar{N}}-\rho_u^{\bar{N}})q_1^{\bar{N}} + \rho_d^{\bar{N}} q_2^{\bar{N}} + p_{exp} q_0^{\bar{N}}) \\
\vdots & \\
q_i^N = \;& (1-\pi)((1-\rho_d^N-\rho_u^N)q_i^N + \rho_d^N q_{i+1}^N + \rho_u^N q_{i-1}^N) \\
& +\pi((1-\rho_d^{\bar{N}}-\rho_u^{\bar{N}})q_i^{\bar{N}} + \rho_d^{\bar{N}} q_{i+1}^{\bar{N}} + \rho_u^{\bar{N}} q_{i-1}^{\bar{N}}) \\
\vdots & \\
q_n^N = \;& (1-\pi)((1-\rho_d^N)q_n^N + \rho_u^N q_{n-1}^N) \\
& +\pi((1-\rho_d^{\bar{N}})q_n^{\bar{N}} + \rho_u^{\bar{N}} q_{n-1}^{\bar{N}}).
\end{aligned} \tag{4.2}$$

These equations seem difficult to solve exactly. But we can get very good approximate solutions. Define:

$$\begin{aligned}
\gamma_i^N = \;& \pi((1-\rho_d^{\bar{N}}-\rho_u^{\bar{N}})q_i^{\bar{N}} + \rho_d^{\bar{N}} q_{i+1}^{\bar{N}} + \rho_u^{\bar{N}} q_{i-1}^{\bar{N}}) \\
& -\pi((1-\rho_d^N-\rho_u^N)q_i^N + \rho_d^N q_{i+1}^N + \rho_u^N q_{i-1}^N) \\
& \text{for } i = 2, \ldots, n; \\
\gamma_1^N = \;& \pi((1-\rho_d^{\bar{N}}-\rho_u^{\bar{N}})q_i^{\bar{N}} + \rho_d^{\bar{N}} q_{i+1}^{\bar{N}} + p_{exp} \\
& -\pi((1-\rho_d^N-\rho_u^N)q_i^N + \rho_d^N q_{i+1}^N + p_{exp}); \\
\gamma_0^N = \;& \pi((1-\rho_d^{\bar{N}})q_n^{\bar{N}} + \rho_u^{\bar{N}} q_{n-1}^{\bar{N}}) - \rho_d^{\bar{N}} q_1^{\bar{N}}.
\end{aligned}$$

Note that γ_i^N is essentially a subexpression of q_i^N. Intuitively, γ_i^N is the net probability transferred between states of (i, N) from (or to) states of the form $(j, \bar{N})$ as a result of nature changing from N to $\bar{N}$ or from $\bar{N}$ to N. Let $(\gamma_i^N)^+ = \gamma_i^N$ if $\gamma_i^N > 0$ and 0 otherwise; let $(\gamma_i^N)^- = \gamma_i^N$ if $\gamma_i^N < 0$ and 0 otherwise. Intuitively, $(\gamma_i^N)^+$ is the net probability transferred to (i, n) from states of the form $(j, \bar{N})$ as a result of nature's state changing from $\bar{N}$ to N; similarly, $(\gamma_i^N)^-$ is the net probability transferred from (i, N) to states of the form $(j, \bar{N})$ as a result of nature's state changing from N to $\bar{N}$. Since $\sum_{i=0}^N q_i^N = 1/2$, it is easy to check that

$$\begin{aligned} &\textstyle\sum_{i=0}^n \gamma_i^N = 0; \\ &-\pi/2 \leq \textstyle\sum_{i=0}^n (\gamma_i^N)^- \leq 0 \leq \sum_{i=0}^n (\gamma_i^N)^+ \leq \pi/2. \end{aligned} \tag{4.3}$$

We can now rewrite the equations in (4.2) using the γ_i^N's to get:

$$\begin{aligned} q_0^N = &\ (1 - p_{exp}) q_0^N + \rho_d^N q_1^N + \gamma_0^N \\ q_1^N = &\ (1 - \rho_d^N - \rho_u^N) q_1^N + \rho_d^N q_2^N + p_{exp} q_0^N + \gamma_1^N \\ &\vdots \\ q_i^N = &\ (1 - \rho_d^N - \rho_u^N) q_i^N + \rho_d^N q_{i+1}^N + \rho_u^N q_{i-1}^N + \gamma_i^N \\ &\vdots \\ q_n^N = &\ (1 - \rho_d^N) q_n^N + \rho_u^N q_{n-1}^N + \gamma_n^N. \end{aligned} \tag{4.4}$$

Although γ_i^N is a function, in general, of q_i^N, q_i^N,q_{i-1}^N,$q_i^{\bar{N}}$, $q_i^{\bar{N}}$, and $q_{i-1}^{\bar{N}}$, we can solve (4.4) by treating it as a constants, subject to the constraints in (4.3). This allows us to break the dependency between the equations for $q_0^B, \ldots, q_n^B$ and those for $q_0^G, \ldots, q_n^G$, and solve them separately. This makes the solution *much* simpler.

By rearranging the arguments, we can express q_n^N as a function of only q_{n-1}^N, ρ_u^N, ρ_d^N, and γ_n^N. By then substituting this expression (where the only unknown

is q_{n-1}^N) for q_n^N in the equation for q_{n-1}^N and rearranging the arguments, we can express q_{n-1}^N in terms of q_{n-2}^N (and the constants ρ_u^N, ρ_d^N, γ_n^N, and γ_{n-1}^N). In general, we can compute q_i^N as a function of q_{i-1}^N (and the constants ρ_u^N, ρ_d^N, γ_i^N and γ_{i-1}^N). Doing this, for $2 \leq i \leq n$ we get

$$q_i^N = \frac{1}{\rho_d^N}(\rho_u^N q_{i-1}^N + (\gamma_n^N + \cdots + \gamma_i^N); \tag{4.5}$$

for q_1 we get

$$q_1^N = \frac{1}{\rho_d^N}(p_{exp} q_0^N + (\gamma_n^N + \ldots + \gamma_1^N). \tag{4.6}$$

Note that substituting the expression for q_1^N into the expression for q_0^N in (4.4) gives $q_0^N = q_0^N$, since $\sum_{i=0}^n \gamma_i^N = 0$.

We can now sum these equations to get

$$\sum_{i=1}^n q_i^N = \frac{1}{\rho_d^N}((\sum_{i=1}^n i\gamma_i^N(+\rho_u^N \sum_{i=1}^{n-1} q_i^N + (p_{exp} q_0^N)).$$

Since $\sum_{i=0}^n q_i^N = 1/2$, it follows that

$$\begin{aligned}(1/2 - q_0^N) = \; & \tfrac{1}{\rho_d^N} \textstyle\sum_{i=1}^n i\gamma_i^N + \rho_u^N(1/2 - q_0^N - q_n^N) \\ & +(p_{exp} q_0^N)).\end{aligned} \tag{4.7}$$

Using the equations in (4.4, we can compute q_i^N as a function of q_0^N. We get

$$q_n^N = \frac{(\rho_u^N)^{n-1}}{(\rho_d^N)^n} p_{exp} q_0^N + (\sum_{i=1}^n \gamma_i) \sum_{j=1}^i \frac{(\rho_u^N)^{n-j}}{(\rho_d^N)^{n-j+1}}.$$

Plugging this back into (4.7) and rearranging the arguments gives us the following equation for q_0^N:

$$\begin{aligned}&(1 + \tfrac{p_{exp}((\sigma_N)^n - 1)}{\rho_u^N - \rho_d^N}) q_0^N = \\ &1/2 + \textstyle\sum_{i=1}^n \gamma_i^N \frac{i - \sum_{j=1}^i ((\sigma_N)^{n-j+1}}{\rho_u^N - \rho_d^N}.\end{aligned} \tag{4.8}$$

Moreover, all the terms that are multiplied by γ_i in (4.8) are negative, and, of these, the one multiplied by γ_n is the largest in absolute value. Given the constraints on $(\gamma_i^N)^+$ and $(\gamma_i^N)^-$ in (4.3), this means that we get a lower bound on

q_0^N by setting $\gamma_n^N = \pi/2$, $\gamma_0^N = -\pi/2$, and $\gamma_i^N = 0$ for $i \neq 0, n$. This is quite intuitive: In order to make q_0^N as small as possible, we want all of the transitions from N to $\bar{N}$ to happen when the automaton is in state 0, and all the transitions from $\bar{N}$ to N to happen when the automaton is in state n, since this guarantees that the expected amount of time that the automaton spends in a state $i > 0$ is maximized. Similarly, to make q_0^N as large as possible, we should set $\gamma_0^N = \pi/2$, $\gamma_n^N = -\pi/2$, and $\gamma_i^N = 0$ for $i \neq 0, N$.

Making these choices and doing some algebra, we get that

$$q_0^N \geq \frac{1}{2}\left(\frac{(\rho_u^N - \rho_d^N) - \pi(\sum_{i=1}^n \sigma_N^i - n)}{(\rho_u^N - \rho_d^N) + p_{exp}((\sigma_N^i)^n - 1)}\right)$$
$$q_0^N \leq \frac{1}{2}\left(\frac{(\rho_u^N - \rho_d^N) + \pi(\sum_{i=1}^n \sigma_N^i - n)}{(\rho_u^N - \rho_d^N) + p_{exp}((\sigma_N^i)^n - 1)}\right).$$

As we have observed before, $E_\pi[A[n, p_{exp}, Pos, Neg, r_u, r_d]] = (1/2 - q_0^G)x_G + (1/2 - q_0^B)x_B$. Plugging in the upper bound for q_0^G and the lower bound for q_0^B gives us the required estimate for Proposition 4.3.3, and completes the proof.

4.4 Experimental Results

In the first part of this section, we examine the performance of $A[n, p_{exp}, Pos, Neg, r_u, r_d]$ with n finite. Using our theoretical analysis, we come up with an estimate of the performance of the automaton, and show that our theoretical estimate is very close to what we observe in simulation. In the second part of the section, we show that the ideas underlying our automaton can be generalized in a natural way to a setting where nature has more than two possible states.

4.4.1 Two states of nature

We focus mostly on scenarios where $\pi = 0.001$, as when nature changes too often, learning from the signals is meaningless (although even for a larger value of π, with a strong enough signal, we can get quite close to the optimal payoff; with smaller π the problem is easier). For simplicity, we also consider an example where $|x_B| = |x_G|$ (we used $x_G = 1, x_B = -1$, but the results would be identical for any other choice of values). We discuss below how this assumption influences the results.

Again, for definiteness, we assume that there are four signals, $1, \ldots, 4$, which have probabilities 0.4, 0.3, 0.2, and 0.1, respectively, when the state of nature is G, and probabilities 0.1, 0.2, 0.3, and 0.4, respectively, when the state of nature is bad. We choose signal 1 to be the "good" signal (i.e., we take $Pos = \{1\}$), and take signal 4 to be the "bad" signal (i.e., we take $Neg = \{4\}$), and take $r_u = r_d = 1$. We ran the process for 10^8 rounds (although the variance was already quite small after 10^6 rounds, and we got good payoff even with 10^5, which is approximately 100 switches between states), using a range of p_{exp} values, and took the result of the best one. We call this $p_{exp}^{opt}(n)$. As can be seen in Figure 4.2, the automaton $A[n, p_{exp}^{opt}(n), \{1\}, \{4\}, 1, 1]$ does well even for small values of n. The optimal expected payoff for an agent that knows nature's state is 0.5. With 4 states, the automaton already gets an expected payoff of more than 0.4; even with 2 states, it gets an expected payoff of more than 0.15.

We also compared the simulation results to the lower bound given by Proposition 4.3.3 (the "est" line in Figure 4.2). As can be seen, the lower bound gives an excellent estimate of the true results. This is actually quite intuitive. The worst-case analysis assumes that all transitions from B to G happen when the

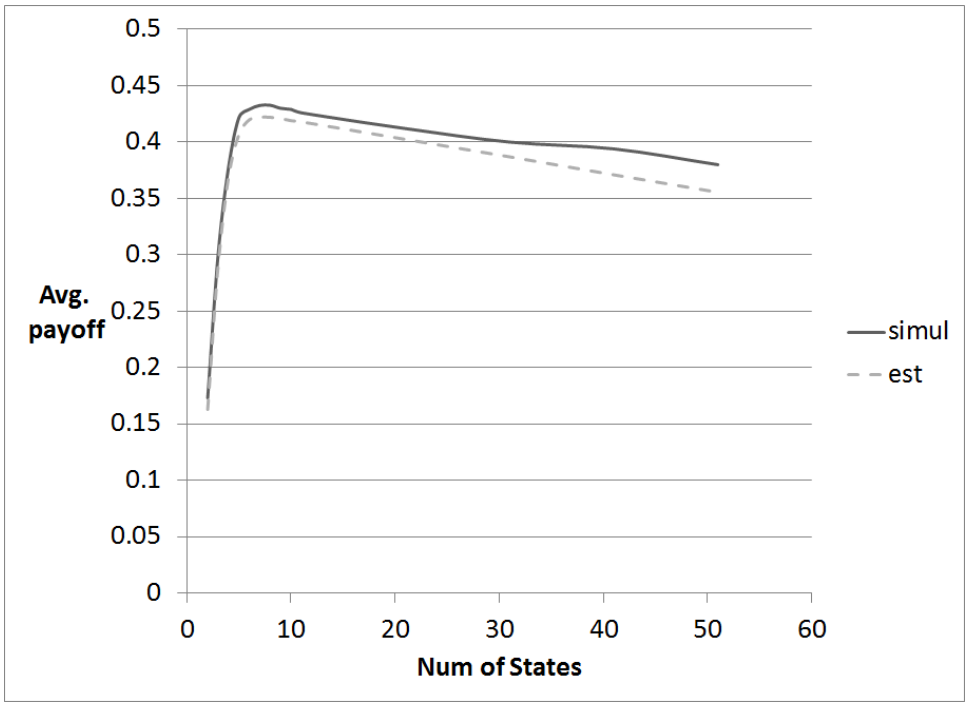

Figure 4.2: Average payoff as a function of the number of states

automaton is in state 0, and all transitions from G to B happen when the automaton is in state n. But when nature is in state B, a "good" automaton should spend most of its time in state 0; similarly, when nature is in state G, a "good" automaton should spend most of its time in state n (as a result of getting good signals). Thus, the assumptions of the worst-case analysis are not far from what we would expect of a good automaton.

Equation (4.1) suggests that while nature is in state G, as the number of states grow, the loss term (that is, the minus term in the $x_G/2$ factor) decreases rapidly. The exact rate of decrease depends on σ_G. We can think of σ_G as describing the quality of the signals that the automaton pays attention to (those in Pos and Neg) when nature is in state G. From equation (4.1), we see that as the number of states grows this loss reduces to $\frac{\pi\sigma_G}{p_{exp}(\sigma^G-1)}$. So the agent's optimal choice is to set the parameters of the automaton so that the ratio is as large as possible. This

allows him to both decrease the loss as fast as possible (with regards to number of states he needs) and to get to the minimal loss possible.

There is of course a tradeoff between σ_G and σ_B. The loss while nature is in state B also decreases rapidly with the number of states, and the rate is dependent on $1/\sigma_B$. As the number of states grows this loss reduces to $\frac{p_{exp}+\pi(\frac{\sigma_B}{1-\sigma_B}-n)}{p_{exp}+\rho_d^B-\rho_u^B}$.

The graph also shows that, somewhat surprisingly, having too many states can hurt, if we fix Pos, Neg, r_u, and r_d. The lower bound in (4.1) actually bears this out. The reason that more states might hurt is that, after a long stretch of time with nature being in state G, the automaton will be in state n. Then if nature switches to state B, it will take the automaton a long time to get to state 0. All this time, it will get a payoff of -1. (Of course, if we allowed the automaton a wider range of strategies, including not using some states, then having more states can never hurt. But we are considering only automata of the form $A[n, p_{exp}, Pos, Neg, r_u, r_d]$.) In a sense, this can be viewed as an explanation of the *recency bias* observed in real decision makers—the tendency to place significantly more weight on recent observations. While the recency bias has often been viewed as inappropriate, these experiments can show that it can be helpful. With more states, more can be remembered, and it becomes harder to "convince" the automaton to change its mind when nature's state actually changes. Having less memory, and thus being more easily influenced by the last few signals, may be more adaptive. By way of contrast, in Wilson's [69] model, nature is static. The optimal automaton in Wilson's setting displayed a strong *first-impression* bias: early observations were extremely influential in the final outcome, rather than recent observations.

We can do a little better by decreasing r_u, thus increasing the amount of time

it will take the automaton to get to state n when nature is in state G. While this helps a little, the effect is not great. Moreover, we do not think it is reasonable to expect resource-bounded agents to "fine-tune" the value of parameters depending on how many states they are willing to devote to a problem. Fortunately, as our experimental results show, they do not need to do such fine-tuning for a wide range of environment settings. There is another tradeoff when choosing the value of p_{exp}, which lies at the heart of the *exploration-exploitation* dilemma. Clearly, if the automaton is in state 0 and nature is in state G, the automaton wants to explore (i.e., move to state 1 and play R) so as to learn that nature is in state G. There are two reasons that the automaton could be in state 0 while nature is in state G. The first is that the automaton gets a sequence of "bad" signals when nature is in state G that force it to state 0. Clearly this is less likely to happen the more states the automaton has. The second is that nature may have switched from B to G while the automaton was in state 0.

Since nature switches from B to G with probability π, a first cut at the exploration probability might be π. However, this first cut is too low an estimate for two reasons. First, the fewer states an automaton has, the more sensitive it is to "bad" signals. Thus, the fewer states an automaton has, the more it should explore. Second, the cost of exploring while nature is in state B is small in comparison to the gain of exploring and discovering out nature has switched to state G. Again, this suggests an increase in the exploration probability. Indeed, we observe that as π gets smaller the optimal p_{exp} value gets smaller, but not in the same ratio. The optimal agent explores less, but still chooses p_{exp} higher than π. For example, with $n = 6$, when changing π from 0.001 to 0.0001 the optimal p_{exp} only changed to from 0.03 to 0.008.

In our simulation, we chose the optimal value of p_{exp} relative to the number of states; this value did not change significantly as a function of the number of states or of the signal profiles. For example, taking $p_{exp} = 0.03$ resulted in payoffs very similar to those with the optimal value of p_{exp} for all $n \geq 5$, and for a wide range of signal profiles while fixing n to 6. This robustness supports our contention that agents do not typically need to fine tune parameter values.

4.4.2 More states of nature

We now consider a setting where nature can have more than two states. Specifically, we allow nature to have $t+1$ states, which we denote $B, G_1, G_2, \ldots, G_t$. In each state, there is probability of π of transitioning to any other state. Again, we have k signals, and the probability of observing signal i is state dependent. The agent has $t+1$ available actions $\{S, E_1, E_2, \ldots, E_t\}$. As before, S is the "safe" action; playing S gives the agent a payoff 0, but also results in the agent receiving no signal. Playing E_i if the state of nature is B result in a payoff of $x_B < 0$; playing E_i when the state of nature is G_i gives the agent a payoff of $x_G > 0$; playing E_i when the state of nature is G_j for $i \neq j$ gives a payoff of 0.

We generalize the family of automata we considered earlier as follows. The family we consider now consists of product automata, with states of the form $(s_0, s_1, \ldots, s_t)$. Each s_i takes on an integer value from 0 to some maximum n. Intuitively, the s_0 component keeps track of whether nature is in state B or in some state other than B; the s_i component keeps track of how likely the state is to be G_i. If $s_0 = 0$, then the automaton plays safe, as before. Again, if $s_0 = 0$, then with probability p_{exp} the automaton explores and changes s_0 to 1. If $s_1 > 0$,

then the automaton plays the action corresponding to the state of nature G_i for which s_i is greatest (with some tie-breaking rule).

We did experiments using one instance of this setting, where nature was in one of five possible states—4 good states and one bad state—and there were six possible signals. We assumed that there was a signal p_i that was "good" for state G_i: it occurred with probability $.6$ when the state of nature was G_i; in state G_j with $j \neq i$, p_i occurred with probability 0.08; similarly, there was a signal that was highly correlated with state B. We considered an automaton for the agent where each of component of the product had five states (so that there were $5^5 = 3125$ states in the automaton. In this setting, the optimal payoff is 0.8. The automaton performed quite well: it was able to get a payoff of 0.7 for an appropriate setting of its parameters.

Part III

Models for computationally bounded agents

CHAPTER 5

THE TRUTH BEHIND THE MYTH OF THE FOLK THEOREM

5.1 Introduction

The complexity of finding a Nash equilibrium (NE) is a fundamental question at the interface of game theory and computer science. A celebrated sequence of results showed that the complexity of finding a NE in a normal-form game is PPAD-complete [11, 12], even for 2-player games. Less restrictive concepts, such as ϵ-NE for an inverse-polynomial ϵ, are just as hard [10]. This suggests that these problems are computationally intractable.

There was some hope that the situation would be better in infinitely-repeated games. The *Folk Theorem* (see [56] for a review) informally states that in an infinitely-repeated game G, for any payoff profile that is *individually rational*, in that all players get more than[1] their minimax payoff (the highest payoff that a player can guarantee himself, no matter what the other players do) and is the outcome of some correlated strategy in G, there is a Nash equilibrium of G with this payoff profile. With such a large set of equilibria, the hope was that finding one would be less difficult. Indeed, Littman and Stone [50] showed that these ideas can be used to design an algorithm for finding a NE in a two-player repeated game.

Borgs et al. [8] (BC+ from now on) proved some results suggesting that, for more than two players, even in infinitely-repeated games it would be difficult to find a NE. Specifically, they showed that, under certain assumptions, the

[1] For our results, since we consider ϵ-NE, we can replace "more than" by "at least".

problem of finding a NE (or even an ϵ-NE for an inverse-polynomial ϵ) in an infinitely repeated game with three or more players where there is a discount factor bounded away from 1 by an inverse polynomial is also PPAD-hard. They prove this by showing that, given an arbitrary normal-form game G with $c \geq 2$ players, there is a game G' with $c + 1$ players such that finding an $\epsilon/8c$-NE for the repeated game based on G' is equivalent to finding an ϵ-NE for G.

While their proof is indeed correct, we challenge their conclusion. If we take seriously the importance of being able to find an ϵ-NE efficiently, it is partly because we have computationally bounded players in mind. But then it seems reasonable to see what happens if we assume that the players in the game are themselves computationally bounded. Like BC+, we assume that players are resource bounded.[2] Formally, we view players as probabilistic[3] polynomial-time Turing machines (PPT TMs). We differ from BC+ in two key respects. First, since we restrict to (probabilistic) polynomial-time players, we restrict the deviations that can be made in equilibrium to those that can be computed by such players; BC+ allow arbitrary deviations. Second, BC+ implicitly assume that players have no memory: they cannot remember computation from earlier rounds. By way of contrast, we allow players to have a bounded (polynomial) amount of memory. This allows players to remember the results of a few coin tosses from earlier rounds, and means that we can use some cryptography (making some standard cryptographic assumptions) to try to coordinate the players. We stress that this coordination happens in the process of the game play, not through

[2]Although BC+ do not discuss modeling players in this way, the problem they show is NP-Hard is to find a polynomial-time TM profile that implements an equilibrium. There is an obvious exponential-time TM profile that implements an equilibrium: each TM in the profile just computes the single-shot NE and plays its part repeatedly.

[3]BC+ describe their TMs as deterministic, but allow them to output a mixed strategy. As they point out, there is no difference between this formulation and a probabilistic TM that outputs a specific action; their results hold for such probabilistic TMs as well.

communication. That is, there are no side channels; the only form of "communication" is by making moves in the game. We call such TMs *stateful*, and the BC+ TMs *stateless*. We note, that without the restriction on deviations, there is no real difference between stateful TMs and stateless TMs in our setting (since a player with unbounded computational power can recreate the necessary state). With these assumptions (and the remaining assumptions of the BC+ model), we show that in fact an ϵ-NE in an infinitely-repeated game can be found in polynomial time.

Our equilibrium strategy uses threats and punishment much in the same way that they are used in the Folk Theorem. However, since the players are computationally bounded we can use cryptography (we assume the existence of a secure public key encryption scheme) to secretly correlate the punishing players. This allows us to overcome the difficulties raised by BC+. Roughly speaking, the ϵ-NE can be described as proceeding in three stages. In the first stage, the players play a sequence of predefined actions repeatedly. If some player deviates from the sequence, the second stage begins, in which the other players use their actions to secretly exchange a random seed, through the use of public-key encryption. In the third stage, the players use a correlated minimax strategy to punish the deviator forever. To achieve this correlation, the players use the secret random seed as the seed of a pseudorandom function, and use the outputs of the pseudorandom function as the source of randomness for the correlated strategy. Since the existence of public-key encryption implies the existence of pseudorandom functions, the only cryptographic assumption needed is the existence of public-key encryptions—one of the most basic cryptographic hardness assumptions.

In the second part of this section we show how to extend this result to a more refined solution concept. While NE has some attractive features, it allows some unreasonable solutions. In particular, the equilibrium might be obtained by what are arguably empty threats. This actually happens in our proposed NE (and in the basic version of the folk theorem). Specifically, players are required to punish a deviating player, even though that might hurt their payoff. Thus, if a deviation occurs, it might not be the best response of the players to follow their strategy and punish; thus, such a punishment is actually an empty threat.

To deal with this (well known) problem, a number of refinements of NE have been considered. The one typically used in dynamic games of perfect information is *subgame-perfect equilibrium*, suggested by Selten [63]. A strategy profile is a subgame-perfect equilibrium if it is a NE at every subgame of the original game. Informally, this means that at any history of the game (even those that are not on any equilibrium path), if all the players follow their strategy from that point on, then no player has an incentive to deviate. In the context of repeated games where players' moves are observed (so that it is a game of perfect information), the folk theorem continues to hold even if the solution concept used is subgame-perfect equilibrium [2, 18, 60].

We define a computational analogue of subgame-perfect equilibrium that we call *computational subgame-perfect ϵ-equilibrium*, where the strategies involved are polynomial-time, and deviating players are again restricted to using polynomial-time strategies. There are a number of subtleties that arise in defining this notion. While we assume that all actions in the underlying repeated game are observable, we allow our TMs to also have memory, which means the action of a TM does not depend only on the public history. Like subgame-

perfect equilibrium, our computational solution concept is intended to capture the intuition that the strategies are in equilibrium after any possible deviation. This means that in a computational subgame-perfect equilibrium, at each history for player i, player i must make a (possibly approximate) best response, no matter what his and the other players' memory states are.

To compute a computational subgame-perfect ϵ-equilibrium, we use the same basic strategy as for NE, but, as often done to get a subgame-perfect equilibrium (for example see [18]), we limit the punishment phase length, so that the players are not incentivized not to punish deviations. However, to prove our result, we need to overcome one more significant hurdle. When using cryptographic protocols, it is often the case (and, specifically is the case in the protocol used for NE) that player i chooses a secret (e.g., a secret key for a public-key encryption scheme) as the result of some randomization, and then releases some public information which is a function of the secret (e.g., a public key). After that public information has been released, another party j typically has a profitable deviation by switching to the TM M that can break the protocol—for every valid public information, there always *exists* some TM M that has the secret "hardwired" into it (although there may not be an efficient way of finding M given the information). We deal with this problem by doing what is often done in practice: we do not use any key for too long, so that j cannot gain too much by knowing any one key.

A second challenge we face is that in order to prove that our new proposed strategies are even an ϵ-NE, we need to show that the payoff of the *best response* to this strategy is not much greater than that of playing the strategy. However, since for any polynomial-time TM there is always a better polynomial-time TM

that has just a slightly longer running time, this natural approach fails. This instead leads us to characterize a class of TMs we can analyze, and show that any other TM can be converted to a TM in this class that has at least the same payoff. While such an argument might seem simple in the traditional setting, since we only allow for polynomial time TMs, in our setting this turns out to require a surprisingly delicate construction and analysis to make sure this converted TM does indeed has the correct size and running time.

The idea of using the structure of the game as a means of correlation is used by Lehrer [49] to show an equivalence between NE and correlated equilibrium in certain repeated games with nonstandard information structures. The use of cryptography in game theory goes back to Urbano and Vila [66, 67], who also used it to achieve coordination between players. More recently, it has been used by, for example, Dodis, Halevi, and Rabin [14].

The application of cryptography perhaps most closely related to ours is by Gossner [24], who uses cryptographic techniques to show how any payoff profile that is above the players' *correlated* minimax value can be achieved in a NE of a repeated game with public communication played by computationally bounded players. In [25], a strategy similar to the one that we use is used to prove that, even without communication, the same result holds. Gossner's results apply only to infinitely-repeated games with 3 players and no discounting; he claims that his results do not hold for games with discounting. Gossner does not discuss the complexity of finding a strategy of the type that he shows exists.

Recently, Andersen and Conitzer [1] described an algorithm for finding NE in repeated games with more than two players with high probability in *uniform games*. However, this algorithm is not guaranteed to work for all games, and

uses the limit of means as its payoff criterion, and not discounting.

There are a few recent papers that investigate solution concepts for extensive-form games involving computationally bounded player [47, 26, 28]; some of these focus on cryptographic protocols [47, 26]. Kol and Naor [47] discuss refinements of NE in the context of cryptographic protocols, but their solution concept requires only that on each history on the equilibrium path, the strategies from that point on form a NE. Our requirement for the computational subgame-perfect equilibrium is much stronger. Gradwohl, Livne and Rosen [26] also consider this scenario and offer a solution concept different from ours; they try to define when an empty threat occurs, and look for strategy profiles where no empty threats are made. Again, our solution concept is much stronger.

5.2 Preliminaries

5.2.1 Infinitely repeated games

Given a normal-form game G[4], we define the repeated game $G^t(\delta)$ as the game in which G is played repeatedly t times (in this context, G is called the *stage game*) and $1-\delta$ is the discount factor (see below). Let $G^\infty(\delta)$ be the game where G is played infinitely many times. An infinite history h in this game is an infinite sequence $\langle \vec{a}^0, \vec{a}^1, \ldots \rangle$ of action profiles. Intuitively, we can think of $\vec{a}^t$ as the action profile played in the t^{th} stage game. We often omit the δ in $G^\infty(\delta)$ if it is not relevant to the discussion. Let H_{G^∞} be the set of all possible histories of G^∞.

[4]To simplify the presentation, we assume all the payoffs in G are normalized so that each player's minimax value is 0. Since, in an equilibrium, all players get at least their minimax value, this guarantees that all players get at least 0 in a correlated equilibrium.

For a history $h \in H_{G^\infty}$ let $G^\infty(h)$ be the subgame that starts at history h (after $|h|$ one-shot games have been played where all players played according to h). We assume that G^∞ is *fully observable*, in the sense that, after each stage game, the players observe exactly what actions the other players played.

A (behavioral) strategy for player i in a repeated game is a function σ from histories of the games to $\Delta(A_i)$. Note that a profile $\vec{\sigma}$ induces a distribution $\rho_{\vec{\sigma}}$ on infinite histories of play. Let $\rho^t_{\vec{\sigma}}$ denote the induced distribution on H^t, the set of histories of length t. (If $t = 0$, we take H^0 to consist of the unique history of length 0, namely $\langle\,\rangle$.) Player i's utility if $\vec{\sigma}$ is played, denoted $p_i(\vec{\sigma})$, is defined as follows:

$$p_i(\vec{\sigma}) = \delta \sum_{t=0}^{\infty} (1-\delta)^t \sum_{h \in H^t, \vec{a} \in A} \rho^{t+1}_{\vec{\sigma}}(h \cdot \vec{a})[u_i(\vec{a})].$$

Thus, the discount factor is $1-\delta$. Note that the initial δ is a normalization factor. It guarantees that if $u_i(\vec{a}) \in [b_1, b_2]$ for all joint actions $\vec{a}$ in G, then i's utility is in $[b_1, b_2]$, no matter which strategy profile $\vec{\sigma}$ is played.

In these game, a more robust solution concept is subgame-perfect equilibrium [63], which requires that the strategies form an ϵ-NE at every history of the game.

Definition 5.2.1 *A strategy profile $\vec{\sigma} = (\sigma_1, ..., \sigma_c)$, is a subgame-perfect ϵ-equilibrium of a repeated game G^∞, if, for all players $i \in [c]$, all histories $h \in H_{G^\infty}$ where player i moves, and all strategies σ' for player i,*

$$p^h_i((\sigma')^h, \vec{\sigma}^h_{-i}) \leq p^h_i(\vec{\sigma}^h) + \epsilon,$$

where p^h_i is the utility function for player i in game $G^\infty(h)$, and σ^h is the restriction of σ to $G^\infty(h)$.

5.2.2 Cryptographic definitions

For a probabilistic algorithm A and an infinite bit string r, $A(x;r)$ denotes the output of A running on input x with randomness r; $A(x)$ denotes the distribution on outputs of A induced by considering $A(x;r)$, where r is chosen uniformly at random. A function $\epsilon : \mathbb{N} \to [0,1]$ is *negligible* if, for every constant $c \in \mathbb{N}$, $\epsilon(k) < k^{-c}$ for sufficiently large k.

We use a *non-uniform* security model, which means our attackers are *non-uniform* PPT algorithm.

Definition 5.2.2 *A* non-uniform *probabilistic polynomial-time machine A is a sequence of probabilistic machines $A = \{A_1, A_2, ...\}$ for which there exists a polynomial d such that both $|A_n|$, the* description size of A_n *(i.e., the states and transitions in A_n), and the running time of A_n are less than $d(i)$.*

Alternatively, a non-uniform PPT machine can also be defined as a uniform PPT machine that receives an advice string (for example, on an extra "advice" tape) for each input length. It is common to assume that the cryptographic building blocks we define next and use in our constructions are secure against non-uniform PPT algorithms.

Computational Indistinguishability

Definition 5.2.3 *A* probability ensemble *is a sequence $X = \{X_n\}_{n\in\mathbb{N}}$ of probability distribution indexed by $\mathbb{N}$. (Typically, in an ensemble $X = \{X_n\}_{n\in\mathbb{N}}$, the support of X_n consists of strings of length n.)*

We now recall the definition of computational indistinguishability [22].

Definition 5.2.4 *Two probability ensembles* $\{X_n\}_{n\in\mathbb{N}}, \{Y_n\}_{n\in\mathbb{N}}$ *are (non-uniformly)* computationally indistinguishable *if, for all (non-uniform) PPT TMs* D*, there exists a negligible function* ϵ *such that, for all* $n \in \mathbb{N}$*,*

$$|\Pr[D(1^n, X_n) = 1] - \Pr[D(1^n, Y_n) = 1]| \leq \epsilon(n).$$

To explain the $\Pr$ *in the last line, recall that* X_n *and* Y_n *are probability distributions. Although we write* $D(1^n, X_n)$*,* D *is a randomized algorithm, so what* $D(1^n, X_n)$ *returns depends on the outcome of random coin tosses. To be a little more formal, we should write* $D(1^n, X_n, r)$*, where* r *is an infinitely long random bit string (of which* D *will only use a finite initial prefix). More formally, taking* $\Pr_{X_n}$ *to be the joint distribution over strings* (x, r) *where* x *is chosen according to* X_n *and r is chosen according to the uniform distribution on bit-strings, we want*

$$|\Pr_{X_n}[\{(x, r) : D(1^n, x, r) = 1\}] - \Pr_{Y_n}[\{(y, r) : D(1^n, y, r) = 1\}]| \leq \epsilon(n).$$

We similarly abuse notation elsewhere in writing $\Pr$*.*

We often call a TM that is supposed to distinguish between two probability ensembles a *distinguisher*. For the rest of this section when we say computationally indistinguishable we mean the non-uniform version.

Pseudorandom Functions

Definition 5.2.5 *A* function ensemble *is a sequence* $F = \{F_n\}_{n\in\mathbb{N}}$ *of probability distributions such that the support of* F_n *is a set of functions mapping* n*-bit strings to* n*-bit strings. The* uniform function ensemble*, denoted* $H = \{H_n\}_{n\in\mathbb{N}}$*, has* H_n *be the uniform distribution over the set of all functions mapping* n*-bit strings to* n*-bit strings.*

We have the same notion of computational indistinguishability for function ensembles as we had for probability ensembles, only that the distinguisher is now an oracle machine, meaning that it can query the value of the function at any point with one computation step, although it does not have the full description of the function. (See [20] for a detailed description.)

We now define *pseudorandom functions* (see [21]). Intuitively, this is a family of functions indexed by a seed, such that it is hard to distinguish a random member of the family from a truly randomly selected function.

Definition 5.2.6 *A* pseudorandom function ensemble (PRF) *is a set* $\{f_s : \{0,1\}^{|s|} \to \{0,1\}^{|s|}\}_{s\in\{0,1\}^*}$ *such that the following conditions hold:*

- *(easy to compute)* $f_s(x)$ *can be computed by a PPT algorithm that is given* s *and* x*;*
- *(pseudorandom) the function ensemble* $F = \{F_n\}_{n\in\mathbb{N}}$*, where* F_n *is uniformly distributed over the multiset* $\{f_s\}_{s\in\{0,1\}^n}$*, is computationally indistinguishable from* H*.*

We use the standard cryptographic assumption that a family of PRFs exists; this assumption is implied by the existence of one-way functions [35, 21]. We actually require the use of a seemingly stronger notion of a PRF, which requires that an attacker getting access to polynomially many instances of a PRF (i.e., f_s for polynomially many values of s) still cannot distinguish them from polynomially many truly random functions. Nevertheless, as we show next, it follows using a standard "hybrid" argument that any PRF satisfies also this stronger "multi-instance" security notion.

Lemma 5.2.7 *For all polynomials q, if $\{f_s : \{0,1\}^{|s|} \rightarrow \{0,1\}^{|s|}\}_{s\in\{0,1\}^*}$ is a pseudo-random function ensemble, then the ensemble $F^q = \{F_n^1, \ldots, F_n^{q(n)}\}_{n\in\mathbb{N}}$ where, for all i, F_n^i is uniformly distributed over the multiset $\{f_s\}_{s\in\{0,1\}^n}$, is computationally indistinguishable from $H^q = \{H_n^1, \ldots, H_n^{q(n)}\}_{n\in\mathbb{N}}$.*

Proof: Assume for contradiction that the ensembles are distinguishable. This means there exist a polynomial q, a PPT D, and a polynomial p such that for infinitely many n's

$$|Pr[D(1^n, (H_n^1, \ldots, H_n^{q(n)})) = 1] - Pr[D(1^n, (F_n^1, \ldots, F_n^{q(n)})) = 1]| > \frac{1}{p(n)}.$$

For each n, let $T_n^i = (1^n, (H_n^1, \ldots, H_n^{i-1}, F_n^i, \ldots, F_n^{q(n)}))$. We can now describe a PPT D' that distinguishes $\{F_n\}_{n\in\mathbb{N}}$ and $\{H_n\}_{n\in\mathbb{N}}$ for infinitely many n's. First notice that a PPT can easily simulate polynomially many oracle queries to both a truly random function and to a member of F_n. So D' on input $(1^n, X)$ randomly chooses $j \in \{1, \ldots, q(n)\}$ and calls D with input $(1^n, (I^1, \ldots, I^{j-1}, X, J^{j+1}, \ldots, J^{q(n)}))$, where it simulates a query to I_k as a query to a random member of H_n, and a query to J_k as a query to a random member of F_n. (Notice that since D is a PPT, it can make only polynomially many oracle queries to any of the functions, which can be easily simulated). Whenever D makes an oracle query to X, D' makes an oracle query to X, and uses its answer as the answer to D. When D terminates, D' outputs the same value as D.

Now notice that if X is H_n, then the input to D is T_n^j, while if X is F_n, then the input to D is T_n^{j+1}. Thus, $Pr[D'(1^n, H_n) = 1] = \frac{1}{q(n)} \sum_{i=1}^{q(n)} Pr[D(T_n^{i+1}) = 1]$, and

$Pr[D'(1^n, F_n) = 1] = \frac{1}{q(n)} \sum_{i=1}^{q(n)} Pr[D(T_n^i) = 1]$. It follows that

$$\begin{aligned}|Pr[D'(1^n, H_n) = 1] - Pr[D'(1^n, F_n) = 1]| &= \frac{1}{q(n)} |\sum_{i=1}^{q(n)} Pr[D(T_n^{i+1}) = 1] - Pr[D(T_n^i) = 1]| \\ &= \frac{1}{q(n)} |Pr[D(T_n^{q(n)+1}) = 1] - Pr[D(T_n^1) = 1]| \\ &> \frac{1}{q(n)p(n)},\end{aligned}$$

where the last inequality is due to the fact that $T_n^{q(n)+1} = (1^n, (H_n^1, \ldots, H_n^{q(n)}))$ and $T_n^1 = (1^n, (F_n^1, \ldots, F_n^{q(n)}))$. But this means that for any such n, D' can distinguish $F = \{F_n\}_{n\in\mathbb{N}}$ and $H = \{H_n\}_{n\in\mathbb{N}}$ with non-negligible probability, and thus can do that for infinitely many n's. This is a contradiction to the assumption that $\{f_s : \{0,1\}^{|s|} \to \{0,1\}^{|s|}\}_{s\in\{0,1\}^*}$ is a pseudorandom function ensemble. □

Public-key Encryption Schemes

We now define public-key encryption schemes. Such a scheme has two keys. The first is public and used for encrypting messages (using a randomized algorithm). The second is secret and used for decrypting. The keys are generated in such a way that the probability that a decrypted message is equal to the encrypted message is equal to 1. The key generation algorithm takes as input a "security parameter" k that is used to determine the security of the protocols (intuitively, no polynomial-time attacker should be able to "break" the security of the protocol except possibly with a probability that is a negligible function of k).

We now recall the formal definitions of public-key encryption schemes [13, 59, 22].

Definition 5.2.8 *A* public-key encryption scheme *is a triple* $\Pi = (Gen, Enc, Dec)$

of PPT algorithms where (a) Gen takes a security parameter 1^k as input and returns a (public key, private key) pair; (b) Enc takes a public key pk and a message m in a message space $\{0,1\}^k$ as input and returns a ciphertext $Enc_{pk}(m)$; (c) Dec is a deterministic algorithm that takes a secret key sk and a ciphertext $\mathcal{C}$ as input and outputs $m' = Dec_{sk}(\mathcal{C})$, and (d)

$$\Pr\left[\exists m \in \{0,1\}^k \textit{ such that } Dec_{sk}(Enc_{pk}(m)) \neq m\right] = 0.$$

We next define a security notion for public-key encryption. Such a security notion considers an adversary that is characterized by two PPT algorithms, A_1 and A_2. Intuitively, A_1 gets as input a public key that is part of a (public key, secret key) pair randomly generated by Gen, together with a security parameter k. A_1 then outputs two messages in $\{0,1\}^k$ (intuitively, messages it can distinguish), and some side information that it passes to A_2 (intuitively, this is information that A_2 needs, such as the messages chosen). A_2 gets as input the encryption of one of those messages and the side information passed on by A_1. A_2 must output which of the two messages m_0 and m_1 the encrypted message is the encryption of (where an output of $b \in \{0,1\}$ indicates that it is m_b). Since A_1 and A_2 are PPT algorithms, the output of A_2 can be viewed as a probability distribution over $\{0,1\}$. The scheme is secure if the two ensembles (i.e., the one generated by this process where the encryption of m_0 is always given to A_2, and the one where the encryption of m_1 is always given to A_2) are indistinguishable. More formally:

Definition 5.2.9 (Public-key security) *A public-key encryption scheme $\Pi = (Gen, Enc, Dec)$ is* secure *if, for every probabilistic polynomial-time adversary $A = (A_1, A_2)$, the ensembles $\{IND_0^\Pi(A,k)\}_k$ and $\{IND_1^\Pi(A,k)\}_k$ are computationally indistinguishable, where $\{IND_b^\Pi(A,k)\}_k$ is the following PPT algorithm:*

$$
\begin{aligned}
\mathit{IND}_b^{\Pi}(A,k) := \quad & (pk, sk) \leftarrow \mathit{Gen}(1^k) \\
& (m_0, m_1, \tau) \leftarrow A_1(1^k, pk)\ (m_0, m_1 \in \{0,1\}^k) \\
& \mathcal{C} \leftarrow \mathit{Enc}_{pk}(m_b) \\
& o \leftarrow A_2(\mathcal{C}, \tau) \\
& \textit{Output } o.
\end{aligned}
$$

Intuitively, the $\leftarrow$ above functions as an assignment statement, but it is not quite that, since the various algorithms are actually PPT algorithms, so their output is randomized. Formally, $\mathit{IND}_b^{\Pi}(A,k)$ is a probability distribution, which we can write as $\mathit{IND}_b^{\Pi}(A,k,r_1,r_2,r_3,r_4)$, where we view r_1, r_2, r_3, and r_4 as the random bitstrings that serve as the second arguments of Gen, A_1, Enc_{pk}, and A_2, respectively. Once we add these arguments (considering, e.g., $\mathit{Gen}(1^k, r_1)$ and $A_1(1^k, pk, r_2)$ rather than $\mathit{Gen}(1^k)$ and $A_1(1^k, pk)$) these algorithms become deterministic, and $\leftarrow$ can indeed be viewed as an assignment statement.

We assume a secure public-key encryption scheme exists. We actually require a seemingly stronger notion of "multi-instance" security, where an attacker gets to see encryptions of multiple messages, each of which is encrypted using multiple keys.

Definition 5.2.10 *A public-key encryption scheme $\Pi = (\mathit{Gen}, \mathit{Enc}, \mathit{Dec})$ is* multi-message multi-key secure *if, for all polynomials f and g, and for every probabilistic polynomial time adversary $A = (A_1, A_2)$, the ensembles $\{\mathit{IND\text{-}M}_0^{\Pi}(A,k,f,g)\}_k$ and $\{\mathit{IND\text{-}M}_1^{\Pi}(A,k,f,g)\}_k$ are computationally indistinguishable, where*

$\mathit{IND\text{-}M}_b^{\Pi}(A, k, f, g) :=$

$(pk_1, sk_1) \leftarrow \mathit{Gen}(1^k), \ldots (pk_{g(k)}, sk_{g(k)}) \leftarrow \mathit{Gen}(1^k),$

$(m_0^1, \ldots, m_0^{f(k)}, m_1^1, \ldots, m_1^{f(k)}, \tau) \leftarrow A_1(1^k, pk_1, \ldots, pk_{g(k)})$ $(m_0^i, m_1^i \in \{0,1\}^k)$

$\mathcal{C} \leftarrow \mathit{Enc}_{pk_1}(m_b^1), \ldots, \mathit{Enc}_{pk_{g(k)}}(m_b^1), \ldots, \mathit{Enc}_{pk_1}(m_b^{f(k)}), \ldots, \mathit{Enc}_{pk_{g(k)}}(m_b^{f(k)})$

$o \leftarrow A_2(\mathcal{C}, \tau)$

Output o

In this definition, there are polynomially many messages being encrypted, and each message is encrypted a polynomial number of times, using a different key each time. Other than that, the process is similar to the standard definition of security. As we show next, any secure encryption scheme is also multi-message multi-key secure.

Lemma 5.2.11 *If* (Gen, Enc, Dec) *is a secure public key encryption scheme, then it is also multi-message multi-key secure.*

Proof: Assume for contradiction that $(\mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec})$ is a secure public key encryption scheme that is not multi-message multi-key secure. Then there exist polynomials f and g and an adversary $A = (A_1, A_2)$ such that $\{\text{IND-M}_0^{\Pi}(A, k, f, g)\}_k$ and $\{\text{IND-M}_1^{\Pi}(A, k, f, g)\}_k$ are distinguishable. That means there exist a PPT D and a polynomial p such that

$$|Pr[D(1^k, \{\text{IND-M}_0^{\Pi}(A, k, f, g)\}) = 1] - Pr[D(1^k, \{\text{IND-M}_1^{\Pi}(A, k, f, g)\}) = 1]| > \frac{1}{p(n)}.$$

Let $T_{i,j}^{\pi}(A, k, f, g)$ be the following PPT algorithm:

$$
\begin{aligned}
T^{\pi}_{i,j}(A,k,f,g) := \quad & (pk_1, sk_1) \leftarrow \mathsf{Gen}(1^k), \ldots (pk_{g(k)}, sk_{g(k)}) \leftarrow \mathsf{Gen}(1^k), \\
& (m_0^1, \ldots, m_0^{f(k)}, m_1^1, \ldots, m_1^{f(k)}, \tau) \leftarrow A_1(1^k, pk_1, \ldots, pk_{g(k)}) \\
& \mathcal{C} \leftarrow \mathsf{Enc}_{pk_1}(m_0^1), \ldots, \mathsf{Enc}_{pk_{g(k)}}(m_0^1), \\
& \ldots, \mathsf{Enc}_{pk_1}(m_0^j), \ldots, \mathsf{Enc}_{pk_{i-1}}(m_0^j), \mathsf{Enc}_{pk_i}(m_1^j), \ldots \mathsf{Enc}_{pk_{g(k)}}(m_1^j), \\
& \ldots \mathsf{Enc}_{pk_1}(m_1^{f(k)}), \ldots, \mathsf{Enc}_{pk_{g(k)}}(m_1^{f(k)}) \\
& o \leftarrow A_2(\mathcal{C}, \tau) \\
& \text{Output } o.
\end{aligned}
$$

We now define an adversary $A' = (A'_1, A'_2)$, and show that $\{\mathrm{IND}_0^{\Pi}(A', k, f, g)\}_k$ and $\{\mathrm{IND}_1^{\Pi}(A', k, f, g)\}_k$ are not computationally indistinguishable. A'_1 on input $(1^k, pk)$ first chooses $i \in \{1, \ldots, g(k)\}$ uniformly at random. It then generates $g(k) - 1$ random key pairs $(pk_1, sk_1), \ldots, (pk_{i-1}, sk_{i-1}), (pk_{i+1}, sk_{i+1}), \ldots, (pk_{g(k)}, sk_{g(k)})$. It then calls A_1 with input $(1^k, pk_1, \ldots, pk_{i-1}, pk, pk_{i+1}, \ldots, pk_{g(k)})$. After getting A_1's output $M = (m_0^1, \ldots, m_0^{f(k)}, m_1^1, \ldots, m_1^{f(k)}, \tau)$, A'_1 chooses $j \in \{1, \ldots, f(n)\}$ uniformly at random, and returns as its output $(m_0^j, m_1^j, (i, j, pk, pk_1, sk_1, \ldots, pk_{g(k)}, sk_{g(k)}, M))$.

A'_2 on input $(\mathcal{C}, (i, j, pk, pk_1, sk_1, \ldots, pk_{g(k)}, sk_{g(k)}, M))$ constructs input $\mathcal{C}'$ for A_2 by first appending the encryptions of messages $m_0^1 \ldots, m_0^{j-1}$ with all the keys, then appending the encryption of m_0^j with keys $pk_1, \ldots, pk_i$ and then appends $\mathcal{C}$. It then appends the encryption of m_1^j with keys $pk_{i+2}, \ldots, pk_{g(k)}$ and also the encryption of the messages $m_1^{j+1}, \ldots, m_1^{f(k)}$ with each of the keys. It then outputs $A_2(\mathcal{C}', \tau)$. If $\mathcal{C}$ is the encryption of m_j^0 with key pk, then this algorithm is identical to $T^{\pi}_{i+1,j}(A, k, f, g)$ (if $i = g(k)$ then by $T^{\pi}_{i+1,j}$ we mean $T^{\pi}_{1,j+1}$; we use similar conventions elsewhere), while if it is the encryption of m_j^1 with key pk, then the algorithm is identical to $T^{\pi}_{i,j}(A, k, f, g)$.

We claim that D can distinguish $\{\text{IND}_0^\Pi(A', k, f, g)\}_k$ and $\{\text{IND}_1^\Pi(A', k, f, g)\}_k$. Note that

$$Pr[D(1^k, \{\text{IND}_0^\Pi(A', k, f, g)\}) = 1] = \frac{1}{g(k)f(k)} \sum_{j=1}^{f(k)} \sum_{i=1}^{g(k)} Pr[D(1^k, T_{i+1,j}^\pi(A, k, f, g)) = 1]$$

and

$$Pr[D(1^k, \{\text{IND}_1^\Pi(A', k, f, g)\}) = 1] = \frac{1}{g(k)f(k)} \sum_{j=1}^{f(k)} \sum_{i=1}^{g(k)} Pr[D(1^k, T_{i,j}^\pi(A, k, f, g)) = 1].$$

Thus,

$$\begin{aligned}
& |Pr[D(1^k, \{\text{IND}_0^\Pi(A', k, f, g)\}) = 1] - Pr[D(1^k, \{\text{IND}_1^\Pi(A', k, f, g)\}) = 1]| \\
= \; & \tfrac{1}{g(k)f(k)} |\textstyle\sum_{j=1}^{f(k)} \sum_{i=1}^{g(k)} (Pr[D(1^k, T_{i+1,j}^\pi(A, k, f, g)) = 1] - Pr[D(1^k, T_{i,j}^\pi(A, k, f, g)) = 1]) \\
= \; & \tfrac{1}{g(k)f(k)} |Pr[D(1^k, \{\text{IND-M}_0^\Pi(A, k, f, g)\}) = 1] - Pr[D(1^k, \{\text{IND-M}_1^\Pi(A, k, f, g)\}) = 1]| \\
> \; & \tfrac{1}{g(k)f(k)p(k)},
\end{aligned}$$

where the next-to-last line follows because $T_{1,1}^\pi(A, k, f, g) = \text{IND-M}_1^\Pi(A, k, f, g)$ and $T_{g(k)+1,f(k)}^\pi(A, k, f, g) = \text{IND-M}_0^\Pi(A, k, f, g)$. Thus, we have a contradiction to the fact that the encryption scheme is secure. □

5.3 The complexity of finding ϵ-NE in repeated games played by stateful machines

5.3.1 Equilibrium Definition

Since we consider computationally-bounded players, we take a player's strategy in G^∞ to be a (possibly probabilistic) Turing machine (TM), which outputs at each round an action to be played, based on its internal memory and the history of play so far. (The TMs considered in BC+ did not have internal memory.)

We consider only TMs that at round t use polynomial in nt many steps to compute the next action, where n is the maximum number of actions a player has in G. Thus, n is a measure of the size of G.[5] Denote by M_i the TM used by player i, and let $\vec{M} = (M_1, \ldots, M_c)$.

We are now ready to define the notion of equilibrium we use. Intuitively, as we model players as polynomial-time TMs, we consider a profile of TMs an equilibrium in a game if there is no player and no other polynomial-time TM that gives that player a higher expected payoff (or up to an ϵ for an ϵ-NE).

Since we consider (probabilistic) TMs that run in polynomial time in the size of the game, we cannot consider a single game. For any fixed game, running in polynomial time in the size of the game is meaningless. Instead, we need to consider a sequence of games. This leads to the following definition.

Definition 5.3.1 *An infinite sequence of strategy profiles $\vec{M}^1, \vec{M}^2, \ldots$, where $\vec{M}^k = (M_1^k, ..., M_c^k)$ is an ϵ-NE of an infinite sequence of repeated games $G_1^\infty, G_2^\infty, \ldots$ where the size of G_k is k if, for all players $i \in [c]$ and all non-uniform PPT adversaries $\bar{M}$ (polynomial in k and t, as discussed above), there exist k_0 such that for all $k \geq k_0$*

$$p_i^k(\bar{M}, \vec{M}_{-i}^k) \leq p_i^k(\vec{M}^k) + \epsilon(k).$$

where p_i^k is the payoff of player i in game G_k^∞.

We note that the equilibrium definition we use considers only deviations that can be implemented by non-uniform polynomial-time TMs. This is different from both the usual definition of NE and from the definition used by BC+, who

[5]When we talk about polynomial-time algorithms, we mean polynomial in n. We could use other measures of the size of G, such as the total number of actions. Since all reasonable choices of size are polynomially related, the choice does not affect our results.

allow arbitrary deviations. But this difference is exactly what allows us to use cryptographic techniques. The need to define polynomial-time deviation is the reason for considering sequences of games instead of a single game. There are other reasonable ways of capturing polynomial-time adversaries. As will be seen from our proof, our approach is quite robust, so our results should hold for any reasonable definition.

5.3.2 Computing an equilibrium

In this section we describe the equilibrium strategy and show how to efficiently compute it. We first start with some definition and lemmas we need for our proof.

Definition 5.3.2 *Let $\mathcal{G}_{a,b,c,n}$ be the set of all games with c players, at most n actions per player, integral payoffs*[6]*, maximum payoff a, and minimum payoff b.*

Note that by our assumption that the minimax payoff is 0 for all players, we can assume $a \geq 0$, $b \leq 0$, and $a - b > 0$ (otherwise $a = b = 0$, which makes the game uninteresting). We start by showing that, given a correlated strategy σ in a game G, players can get an average payoff that is arbitrarily close to their payoff in σ by playing a fixed sequence of action profiles repeatedly.

Lemma 5.3.3 *For all a, b, c, all polynomials q, all n, all games $G \in \mathcal{G}_{a,b,c,n}$, and all correlated strategies σ in G, if the expected payoff vector of playing σ is p then there*

[6]Our result also hold for rational payoffs except then the size of the game needs to take into account the bits needed to represent the payoffs

exists a sequence sq of length $w(n)$, where $w(n) = ((a-b)q(n)+1)n^c$, such that player i's average payoff in sq is at least $p_i - 1/q(n)$.

Proof: Given σ, we create sq the obvious way: by playing each action in proportion to the probability $\sigma(\vec{a})$. More precisely, let $r = a - b$, and define $w(n) = (rq(n) + 1)n^c$, as in the statement of the lemma. We create a sequence sq by playing each action profile $\vec{a}$ $\lfloor w(n)\sigma(\vec{a})\rfloor$ times, in some fixed order. Notice that the length of this sequence is between $w(n) - n^c$ and $w(n)$. The average payoff player i gets in sq is

$$\begin{aligned} v_i' &= \frac{1}{\sum_{\vec{a}\in A}\lfloor w(n)\sigma(\vec{a})\rfloor}\sum_{\vec{a}\in A}\lfloor w(n)\sigma(\vec{a})\rfloor u_i(\vec{a}) \\ &\geq \frac{1}{\sum_{\vec{a}\in A}\lfloor w(n)\sigma(\vec{a})\rfloor}\left(\sum_{\vec{a}\in A, u_i(\vec{a})\geq 0}(w(n)\sigma(\vec{a})-1)u_i(\vec{a}) + \sum_{\vec{a}\in A, u_i(\vec{a})<0} w(n)\sigma(\vec{a})u_i(\vec{a})\right) \\ &= \frac{w(n)\sum_{\vec{a}\in A}\sigma(\vec{a})u_i(\vec{a})}{\sum_{\vec{a}\in A}\lfloor w(n)\sigma(\vec{a})\rfloor} - \frac{\sum_{\vec{a}\in A, u_i(\vec{a})\geq 0} u_i(\vec{a})}{\sum_{\vec{a}\in A}\lfloor w(n)\sigma(\vec{a})\rfloor} \geq \frac{w(n)p_i}{\sum_{\vec{a}\in A}\lfloor w(n)\sigma(\vec{a})\rfloor} - \frac{an^c}{w(n)-n^c}. \end{aligned}$$

If $p_i < 0$,

$$\begin{aligned} v_i' &\geq \frac{w(n)p_i}{\sum_{\vec{a}\in A}\lfloor w(n)\sigma(\vec{a})\rfloor} - \frac{an^c}{w(n)-n^c} \geq \frac{w(n)p_i - an^c}{w(n)-n^c} \\ &= \frac{(rq(n)+1)n^c p_i - an^c}{(rq(n)+1)n^c - n^c} = \frac{rq(n)n^c p_i - (a-p_i)n^c}{rq(n)n^c} \geq p_i - \frac{1}{q(n)}. \end{aligned}$$

If $p_i \geq 0$,

$$\begin{aligned} v_i' &\geq \frac{w(n)p_i}{\sum_{\vec{a}\in A}\lfloor w(n)\sigma(\vec{a})\rfloor} - \frac{an^c}{w(n)-n^c} \geq p_i - \frac{an^c}{w(n)-n^c} \\ &= p_i - \frac{an^c}{(rq(n)+1)n^c - n^c} = p_i - \frac{an^c}{rq(n)n^c} \geq p_i - \frac{1}{q(n)}. \end{aligned}$$

□

Lemma 5.3.4 *For all a, b, c, all polynomials q and w, all $G \in \mathcal{G}_{a,b,c,n}$, and all sequences sq of length $w(n)$, if the average payoff vector of playing sq is p, then for all $\delta \leq 1/f(n)$,*

where $f(n) = (a-b)w(n)q(n)$, if sq is played infinitely often, player i's payoff in $G^\infty(\delta)$ is at least $p_i - 1/q(n)$.

Proof: Suppose that $sq = (a_0, \ldots, a_{w(n)-1})$, and let v_i be i's payoff from sq^∞ in $G^\infty(\delta)$. Then

$$\begin{aligned} v_i &= \delta \sum_{t=0}^{\infty} (1-\delta)^{tw(n)} \sum_{k=0}^{w(n)-1} u(a_k)(1-\delta)^k \\ &= p_i + \delta \sum_{t=0}^{\infty} (1-\delta)^{tw(n)} \sum_{k=0}^{w(n)-1} (u(a_k) - p_i)(1-\delta)^k. \end{aligned}$$

We want to bound the loss from the second part of the sum. Notice that this is a discounted sum of a sequence whose average payoff is 0. Call this sequence sq'. Observe that, because of the discounting, in the worst case, i gets all of his negative payoff in the first round of sq' and all his positive payoffs in the last round. Thus, we can bound the discounted average payoff by analyzing this case. Let the sum of i's negative payoffs in sq' be P_{neg}, which means that the sum of i's positive payoffs must be $-P_{neg}$. Let $r = a - b$, let $v_i' = \min_{\vec{a} \in A}(u_i(\vec{a}) - p_i) \geq -r$, and let $f(n) = rw(n)q(n)$, as in the statement of the lemma. So, if $\delta \leq 1/f(n)$, player $i's$ average discounted payoff in the game is at least

$$\begin{aligned} v_i &\geq p_i + \delta \sum_{t=0}^{\infty} P_{neg}(1-\delta)^{w(n)t} + (-P_{neg})(1-\delta)^{w(n)(t+1)-1} \\ &= p_i + \delta(P_{neg} + (-P_{neg})(1-\delta)^{w(n)-1}) \sum_{t=0}^{\infty} (1-\delta)^{w(n)t} \\ &= p_i + \delta(P_{neg} + (-P_{neg})(1-\delta)^{w(n)-1}) \frac{1}{1-(1-\delta)^{w(n)}} \\ &= p_i + P_{neg}\delta \frac{1-(1-\delta)^{w(n)-1}}{(1-(1-\delta)^{w(n)})} \geq p_i + \delta P_{neg} \geq p_i + \frac{P_{neg}}{f(n)} \geq p_i + \frac{v_i' w(n)}{f(n)} = p_i - 1/q(n). \end{aligned}$$

□

The next lemma shows that, for every inverse polynomial, if we "cut off" the game after some appropriately large polynomial p number of rounds (and

compute the discounted utility for the finitely repeated game considering only $p(n)$ repetitions), each player's utility in the finitely repeated game is negligibly close to his utility in the infinitely repeated game—that is, the finitely repeated game is a "good" approximation of the infinitely repeated game.

Lemma 5.3.5 *For all a, b, c, all polynomials q, all n, all games $G \in \mathcal{G}_{a,b,c,n}$, all $0 < \delta < 1$, all strategy profiles $\vec{M}$, and all players i, i's expected utility $p_i[\vec{M}]$ in game $G^{\lceil n/\delta \rceil}(\delta)$ and $p_i[\vec{M}]$ in game $G^{\infty}(\delta)$ differ by at most a/e^n.*

Proof: Let $p_i^t(\vec{M})$ denote player i's expected utility if the players are playing $\vec{M}$ and the game ends at round t. Recall that $(1-\delta)^{1/\delta} \leq 1/e$.

$$
\begin{aligned}
& p_i^{\infty}(\vec{M}) - p_i^{\lceil n/\delta \rceil}(\vec{M}) \\
= \;& \delta \textstyle\sum_{t=0}^{\infty}(1-\delta)^t \sum_{h \in H^t, \vec{a} \in A} \rho_{\vec{M}}^{t+1}(h \cdot \vec{a})[u_i(\vec{a})] - \delta \sum_{t=0}^{\lceil n/\delta \rceil}(1-\delta)^t \sum_{h \in H^t, \vec{a} \in A} \rho_{\vec{M}}^{t+1}(h \cdot \vec{a})[u_i(\vec{a})] \\
= \;& \delta \textstyle\sum_{t=\lceil n/\delta \rceil + 1}^{\infty}(1-\delta)^t \sum_{h \in H^t, \vec{a} \in A} \rho_{\vec{M}}^{t+1}(h \cdot \vec{a})[u_i(\vec{a})] \\
\leq \;& \delta \textstyle\sum_{t=\lceil n/\delta \rceil}^{\infty}(1-\delta)^t a \\
= \;& \delta(1-\delta)^{\lceil n/\delta \rceil} \textstyle\sum_{t=0}^{\infty}(1-\delta)^t a = \delta(1-\delta)^{\lceil n/\delta \rceil} \frac{a}{\delta} \leq \frac{a}{e^n}.
\end{aligned}
$$

□

The ϵ-NE strategy and the algorithm

Let $A_i^0 \subset A_i$ be a non-empty set and let $A_i^1 = A_i \setminus A_i^0$.[7] A player can broadcast an m-bit string by using his actions for m rounds, by treating actions from A_i^0 as 0 and actions from A_i^1 as 1. Let $(\text{Gen}, \text{Enc}, \text{Dec})$ be a multi-message multi-key secure public-key encryption scheme, such that if the security parameter is k, the length of the public key is $v(k)$ and the length of an encrypted message is

[7] We assume that each player has at least two actions in G. This assumption is without loss of generality—we can essentially ignore players for whom it does not hold.

$z(k)$ for some polynomials v and z. Let $sq = (s_1, s_2 \ldots, s_m)$ be a fixed sequence of action profiles. Fix a polynomial-time pseudorandom function ensemble $\{PS_s : s \in \{0,1\}^*\}$. For a game G such that $|G| = n$, consider the strategy σ^{NE} for player i in $G^\infty(\delta)$ that has the following three phases. Phase 1 explains what to do if no deviation occurs: play sq. Phase 2 gives the preliminaries of what to do if a deviation does occur: roughly, compute a random seed that is shared with all the non-deviating players. Phase 3 explains how to use the random seed to produce a correlated punishment strategy that punishes the deviating player. Formally let $\vec{M}^{\sigma^{NE}}$ be the TMs that implement the following strategy:

1. Play according to sq (with wraparound) as long as all players played according to sq in the previous round.
2. After detecting a deviation by player $j \neq i$ in round t_0:[8]
   (a) Generate a pair (pk_i, sk_i) using Gen(1^n). Store sk_i in memory and use the next $v(n)$ rounds to broadcast pk_i, as discussed above.
   (b) If $i = j + 1$ (with wraparound), player i does the following:
      - i records $pk_{j'}$ for all players $j' \notin \{i, j\}$;
      - i generates a random n-bit string $seed$;
      - for each player $j' \notin \{i, j\}$, i computes $m = Enc_{pk_{j'}}(seed)$, and uses the next $(c-2)z(n)$ rounds to communicate these strings to the players other than i and j (in some predefined order).
   (c) If $i \neq j + 1$, player i does the following:
      - i records the actions played by $j + 1$ at time slots designated for i to retrieve $Enc_{Pk_i}(seed)$;

[8] If more than one player deviates while playing sq, the players punish the one with the smaller index. The punished player plays his best response to what the other players are doing in this phase.

- i decrypts to obtain $seed$, using Dec and sk_i.

3. Phase 2 ends after $v(n) + (c-2)z(n)$ rounds. The players other than j then compute $PS_{seed}(t)$ and use it to determine which action profile to play according to the distribution defined by a fixed (correlated) punishment strategy against j.

Note that if the players other than j had played a punishment strategy against j, then j would get his minimax payoff of 0. What the players other than j are actually doing is playing an approximation to a punishment strategy in two senses: first they are using a psuedorandom function to generate the randomness, which means that they are not quite playing according to the actual punishment strategy. Also, j might be able to guess which pure strategy profile they are actually playing at each round, and so do better than his minimax value. As we now show, j's expected gain during the punishment phase is negligible.

Lemma 5.3.6 *For all a, b, c, all polynomials t and f, all n, and all games $G \in \mathcal{G}_{a,b,c,n}$, in $G^\infty(1/f(n))$, if the players other than j play $\vec{M}_{-j}^{\sigma^{NE}}$, then if j deviates at round $t(n)$, j's expected payoff during the punishment phase is negligible.*

Proof: Since we want to show j's expected payoff during the punishment phase (phase (3) only) is negligible, it suffices to consider only polynomially many rounds of playing phase (3) (more precisely, at most $nf(n)$ rounds); by Lemma 5.3.5, any payoff beyond then is guaranteed to be negligible due to the discounting.

We construct three variants of the strategy $\vec{M}_{-j}^{\sigma^{NE}}$, that vary in phases (2) and (3). We can think of these variants as interpolating between the strategy above

and the use of true randomness. (These variants assume an oracle that provides appropriate information; these variants are used only to make the claims precise.)

H1 In phase (2), the punishing players send their public keys to $j+1$. For each player j' not being punished, player $j+1$ then encrypts the seed 0 using (j')'s public key, and then sends the encrypted key to j'. In phase (3), the punishing players get the output of a truly random function (from an oracle), and use it to play the true punishment strategy. (In this case, phase (2) can be eliminated.)

H2 In phase (2), the punishing players send their public keys to $j+1$. For each player j' not being punished, player $j+1$ encrypts the seed 0 using (j')'s public key, and then sends the encrypted key to j'. In phase (3), the punishing players get a joint random string $seed$ (from an oracle) and use the outputs of PS_{seed} to decide which strategy profile to play in each round. (Again, in this case, phase (2) can be eliminated.)

H3 In phase (2), the punishing players send their public keys to $j+1$. Player $j+1$ chooses a random string $seed$ and, for each player j' not being punished, $j+1$ encrypts $seed$ using (j')'s public key, and then sends the encrypted key to j'. In phase (3), the punishing players use the outputs of PS_{seed} to decide which strategy profile to play in each round.

It is obvious that in $H1$, j's expected payoff is negligible. (Actually, there is a slight subtlety here. As we observed above, using linear programming, we can compute a strategy that gives the correlated minimax, which gives j an expected payoff of 0. To actually implement this correlated minimax, the players need to sample according to the minimax distribution. They cannot

necessarily do this exactly (for example, $1/3$ can't be computed exactly using random bits). However, given n, the distribution can be discretized to the closest rational number of the form $m/2^n$ using at most n random bits. Using such a discretized distribution, the players other than j can ensure that j gets only a negligible payoff.)

We now claim that in $H2$, j's expected payoff during the punishment phase is negligible. Assume for contradiction that a player playing $H2$ has a non-negligible payoff $\mu(n)$ for all n (i.e., there exists some polynomial $g(\cdot)$ such that $\mu(n) \geq 1/g(n)$ for infinitely many n.). Let $h(n) = n(a-b)^2(1/\mu(n))^2$. We claim that if j's expected payoff is non-negligible, then we can distinguish $h(n)$ instances of the PRF $\{PS_s : s \in \{0,1\}^n\}$ with independently generated random seeds, from $h(n)$ independent truly random functions, contradicting the multi-instance security of the PRF PS.

More precisely, we construct a distinguisher D that, given 1^n and oracle access to a set of functions $f^1, f^2, \ldots, f^{h(n)}$, proceeds as follows. It simulates H_2 (it gets the description of the machines to play as its non-uniform advice) $h(n)$ times where in iteration i', it uses the function $f^{i'}$ as the randomization source of the correlated punishment strategy. D then computes the average payoff of player j in the $h(n)$ runs, and outputs 1 if this average exceeds $\mu(n)/2$. Note that if the functions $f^1, f^2, \ldots, f^{h(n)}$ are truly independent random functions, then D perfectly simulates H_1 and thus, in each iteration i', the expected payoff of player j (during the punishment phase) is negligible. On the other hand, if the functions $f^1, f^2, \ldots, f^{h(n)}$ are $h(n)$ independent randomly chosen instances of the PRF $\{PS_s : s \in \{0,1\}^n\}$, then D perfectly simulates H_2, and thus, in each iteration i', the expected payoff of player j (during the punishment phase) is at

least $\mu(n)$.

By Hoeffding's inequality [38], given m random variables $X_1, \ldots, X_m$ all of which take on values in an interval of size c', $p(|\overline{X} - E(\overline{X})| \geq r) \leq 2exp(-\frac{2mr^2}{c'^2})$. Since, in this setting, the range of the random variables is an interval of size $a - b$, the probability that D outputs 1 when the function are truly independent is at most $2/e^n$, while the probability that D outputs 1 when the functions are independent randomly chosen instances of the PRF $\{PS_s : s \in \{0,1\}^n\}$ is at least $1 - 2/e^n$. This, in turn, means that the difference between them is not negligible, which is a contradiction. Thus, j's expected payoff in $H2$ must be negligible.

We now claim that in $H3$, player j's expected payoff during the punishment phase is also negligible. Indeed, if j can get a non-negligible payoff, then we can break the multi-message multi-key secure encryption scheme.

Again, assume for contradiction that the punished player's expected payoff in the punishment phase is a non-negligible function $\mu(n)$ for all n. We can build a distinguisher $A = (A_1, A_2)$ (which also gets the description of the machines to play as its non-uniform advice) to distinguish $\{\text{IND-M}_0^{\Pi}(A, n, h, c)\}_n$ and $\{\text{IND-M}_1^{\Pi}(A, n, h, c)\}_n$ (where we abuse notation and identify c with the constant polynomial that always returns c). Given n, A_1 randomly selects $h(n)$ messages $r_1, \ldots, r_{h(n)}$ and outputs $(0, \ldots, 0, r_1, \ldots, r_{h(n)}, (pk_1, \ldots, pk_c))$. A_2 splits its input into pieces. The first piece contains the first c encryptions in $\mathcal{C}$ (i.e., the c encryptions of the first message chosen, according to the c different encryption functions), the second the next c encryptions and so on. Notice that each piece consists of c different encryptions of the same message in both cases. It can also simulate phase (1) by following the strategy for t rounds. It then uses each piece, along with the public keys, to simulate the communication in phase

(2). For piece j it uses r_j as the seed of the PRF in phase (3). It repeats this experiment for all the different pieces of the input, for a total of $h(n)$ times, and outputs 1 if the punished player's average payoff over all experiments using its strategy is more than $\mu(n)/2$.

Note that if $b = 1$, player j faces $H3$ (i.e., the distributions over runs when $b = 1$ is identical to the distribution over runs with H3, since in both cases the seed is chosen at random and the corresponding messages are selected the same way), so player j's expected payoff in the punishment phase is $\mu(n)$. Thus, by Hoeffding's inequality the probability that player j's average payoff in the punishment phase is more then $\mu(n)/2$ is $1 - 2/e^n$, so A_2 outputs 1 with that probability in the case $b = 1$. On the other hand, if $b = 0$, then this is just H_2. We know player j's expected payoff in the punishment phase in each experiment is no more than negligible in H_2, so the probability that the average payoff is more than $\mu(n)/2$ after $h(n)$ rounds, is negligible. This means that there is a non-negligible difference between the probability A outputs 1 when $b = 1$ and when $b = 0$, which contradicts the assumption that the encryption scheme is multi-message multi-key secure public key secure. Thus, the gain in $H3$ must be negligible.

$H3$ is exactly the game that the punished player faces; thus, this shows he can't hope to gain more than a negligible payoff in expectation. □

We can now state and prove our main theorem, which says that σ^{NE} is an ϵ-NE for all inverse polynomials ϵ and can be computed in polynomial time.

Theorem 5.3.7 *For all a, b, c, and all polynomials q, there is a polynomial f and a polynomial-time algorithm F such that, for all sequences $G_1, G_2, \ldots$ of games with*

$G^j \in G_{a,b,c,j}$ and for all inverse polynomials $\delta \leq 1/f$, the sequence of outputs of F given the sequence $G_1, G_2, \ldots$ of inputs is a $\frac{1}{q}$-NE for $G_1^\infty(\delta(1)), G_2^\infty(\delta(2)), \ldots$.

Proof: Given a game $G^n \in \mathcal{G}(a, b, c, n)$, the first step of the algorithm is to find a correlated equilibrium σ of G^n. This can be done in polynomial time using linear programming. Since the minimax value of the game is 0 for all players, all players have an expected utility of at least 0 using σ. Let $r = a - b$. By Lemma 5.3.3, we can construct a sequence sq of length $w(n) = (3rnq(n)+1)n^c$ that has an average payoff for each player that is at most $1/3q(n)$ less than his payoff using σ. By Lemma 5.3.4, it follows that by setting the discount factor $\delta < 1/f'(n)$, where $f'(n) = 3rw(n)q(n)$, the loss due to discounting is also at most $1/3q(n)$. We can also find a punishment strategy against each player in polynomial time, using linear programming.

We can now compute the strategy $\vec{M}^{\sigma^{NE}}$ described earlier that uses the sequence sq and the punishment strategies. Let $\vec{\sigma}_n^*$ be this strategy when given G_n as input. Let $m(n) = v(n) + (c-2)z(n)$ (the length of phase (2)). Let

$$f(n) = \max(3q(n)(m(n)a + 1), f'(n)).$$

Notice that f is independent of the actual game as required.

We now show that $\vec{\sigma}_1^*, \ldots$ as defined above is a $(1/q)$-NE. If in game G^n a player follows σ_n^*, he gets at least $-2/3q(n)$. Suppose that player j defects at round t; that is, that he plays according to σ_n^* until round t, and then defects. By Lemma 5.3.5 if $t > \frac{n}{\delta(n)}$, then any gain from defection is negligible, so there exists some n_1 such that, for all $n > n_1$, a defection in round t cannot result in the player gaining more than $\frac{1}{q(n)}$. If player j defects at round $t \leq \frac{n}{\delta(n)}$, he gets at most a for the duration of phase (2), which is at most $m(n)$ rounds, and then, by

Lemma 5.3.6, gains only a negligible amount, say $\epsilon_{neg}(n)$ (which may depend on the sequence of deviations), in phase (3). Let u_i^n be the payoff of player i in game G^n of the sequence. It suffices to show that

$$\delta(n)(\sum_{k=0}^{t} u_i^n(a_k)(1-\delta(n))^k + \sum_{k=0}^{m(n)} a(1-\delta(n))^{k+t} + (1-\delta(n))^{t+m(n)}\epsilon_{neg}(n)) - 1/q(n)$$
$$\leq \delta(n)(\sum_{k=0}^{t} u_i^n(a_k)(1-\delta(n))^k + \sum_{k=t}^{\infty} u_i^n(a_k)(1-\delta(n))^k).$$

By deleting the common terms from both side, rearranging, and noticing that $(1-\delta(n))^{m(n)}\epsilon_{neg(n)} \leq \epsilon_{neg(n)}$, it follows that it suffices to show

$$\delta(n)(1-\delta(n))^t(\sum_{k=0}^{m(n)} a(1-\delta(n))^k + \epsilon_{neg}(n)) - \frac{1}{q(n)} \leq$$
$$\delta(n)(1-\delta(n))^t(\sum_{k=0}^{\infty} u_i^n(a_{k+t})(1-\delta(n))^k).$$

We divide both sides of the equation by $(1-\delta(n))^t$. No matter at what step of the sequence the defection happens, the future expected discounted payoff from that point on is still at least $-2/3q(n)$, as our bound applies for the worst sequence for a player, and we assumed that in equilibrium all players get at least 0. It follows that we need to show

$$\delta(n)(\sum_{k=0}^{m(n)} a(1-\delta(n))^k + \epsilon_{neg}(n)) - \frac{1}{q(n)(1-\delta(n))^t} \leq -\frac{2}{3q(n)}.$$

Since ϵ_{neg} is negligible for all deviations, it follows that, for all sequences of deviations, there exists n_0 such that $\epsilon_{neg}(n) < 1$ for all $n \geq n_0$. For $n \geq n_0$,

$$\begin{aligned}
& \delta(n)(\textstyle\sum_{k=0}^{m(n)} a(1-\delta(n))^k + \epsilon_{neg}(n)) - \frac{1}{q(n)(1-\delta(n))^t} \\
\leq\ & \delta(n)(m(n)a + \epsilon_{neg}(n)) - \tfrac{1}{q(n)} \\
\leq\ & \tfrac{m(n)a+\epsilon_{neg}(n)}{f(n)} - \tfrac{1}{q(n)} \\
\leq\ & \tfrac{m(n)a+\epsilon_{neg}(n)}{3q(n)(m(n)a+1)} - \tfrac{1}{q(n)} \\
\leq\ & \tfrac{1}{3q(n)} - \tfrac{1}{q(n)} \\
=\ & -\tfrac{2}{3q(n)}.
\end{aligned}$$

This shows that there is no deviating strategy that can result in the player gaining more than $\frac{1}{q(n)}$ in G^n for $n > \max\{n_0, n_1\}$. □

5.3.3 Dealing with a variable number of players

Up to now, we have assumed, just as in Borgs et al. [8], that the number of players in the game is a fixed constant (≥ 3).

What happens if the number of players in the game is part of the input? In general, describing the players' utilities in such a game takes space exponential in the number of players (since there are exponentially many strategy profiles). Thus, to get interesting computational results, we consider games that can be represented succinctly.

Graphical games [46] of degree d are games that can be represented by a graph in which each player is a node in the graph, and the utility of a player is a function of only his action and the actions of the players to which he is connected by an edge. The maximum degree of a node is assumed to be at most d. This means a player's punishment strategy depends only on the actions of at most d players.

Definition 5.3.8 *Let $\mathcal{G}'_{a,b,d,n,m}$ be the set of all graphical games with degree at most d, at most m players and at most n actions per player, integral payoffs,[9] maximum payoff a, and minimum payoff b.*

The following corollary then follows from the fact that a correlated equilib-

[9] Again, our result also holds for rational payoffs, except then the size of the game needs to take into account the bits needed to represent the payoffs.

rium with polynomial sized-support can be computed in polynomial time [41], the observation that we can easily compute a correlated minimax strategy that depends only on the actions of at most d players and our theorem (where in Lemma 5.3.3 we replace n^c in the definition of $w(n)$ with the size of the support of the correlated equilibrium).

Corollary 5.3.9 *For all a, b, d, and all polynomials q, there is a polynomial f and a polynomial-time algorithm F such that, for all sequences $G_1, G_2, \ldots$ of games with $G^j \in G_{a,b,d,j,j}$ and for all inverse polynomials $\delta \leq 1/f$, the sequence of outputs of F given the sequence $G_1, G_2, \ldots$ of inputs is a $\frac{1}{q}$-equilibrium for $G_1^\infty(\delta(1)), G_2^\infty(\delta(2)), \ldots$.*

5.4 Computational subgame-perfect equilibrium

5.4.1 Motivation and Definition

In this section we would like to define a notion similar to subgame-perfect equilibrium, where for all histories h in the game tree (even ones not on the equilibrium path), playing $\vec{\sigma}$ restricted to the subtree starting at h forms a NE. This means that a player does not have any incentive to deviate, no matter where he finds himself in the game tree.

As we suggested in the introduction, there are a number of issues that need to be addressed in formalizing this intuition in our computational setting. First, since we consider stateful TMs, there is more to a description of a situation than just the history; we need to know the memory state of the TM. That is, if we take a history to be just a sequence of actions, then the analogue of history for

us is really a pair $(h, \vec{m})$ consisting of a sequence h of actions, and a profile of memory states, one for each player. Thus, to be a computational subgame-perfect equilibrium the strategies should be a NE at every history and no matter what the memory states are.

Another point of view is to say that the players do not in fact have perfect information in our setting, since we allow the TMs to have memory that is not observed by the other players, and thus the game should be understood as a game of imperfect information. In a given history h where i moves, i's information set consists of all situations where the history is h and the states of memory of the other players are arbitrary. While subgame-perfect equilibrium extends to imperfect information games it usually doesn't have much bite (see [48] for a discussion on this point). For the games that we consider, subgame-perfect equilibrium typically reduces to NE. An arguably more natural generalization of subgame-perfect equilibrium in imperfect-information games would require that if an information set for player i off the equilibrium path is reached, then player i's strategy is a best response to the other players' strategies *no matter how that information set is reached*. This is quite a strong requirement. (see [56][pp. 219–221] for a discussion of this issue); such equilibria do not in general exist in games of imperfect information.

Instead, in games of imperfect information, the solution concept most commonly used is *sequential equilibrium* [48]. A sequential equilibrium is a pair $(\vec{\sigma}, \mu)$ consisting of a strategy profile $\vec{\sigma}$ and a *belief system* μ, where μ associates with each information set I a probability $\mu(I)$ on the nodes in I. Intuitively, if I is an information set for player i, $\mu(I)$ describes i's beliefs about the likelihood of being in each of the nodes in I. Then $(\vec{\sigma}, \mu)$ is a sequential equilibrium if, for

each player i and each information set I for player i, σ_i is a best response to $\vec{\sigma}_{-i}$ given i's beliefs $\mu(I)$. However, a common criticism of this solution concept is that it is unclear what these beliefs should be and how players create these beliefs. Instead, our notion of computational subgame-perfection can be viewed as a strong version of a sequential equilibrium, where, for each player i and each information set I for i, σ_i is a best response to $\vec{\sigma}_{-i}$ conditional on reaching I (up to ϵ) no matter what i's beliefs are at I.

As a deviating TM can change its memory state in arbitrary ways, when we argue that a strategy profile is an ϵ-NE at a history, we must also consider all possible states that the TM might start with at that history. Since there exists a deviation that just rewrites the memory in the round just before the history we are considering, any memory state (of polynomial length) is possible. Thus, in the computational setting, we require that the TM's strategies are an ϵ-NE at every history, no matter what the states of the TMs are at that history. This solution concept is in the spirit of subgame-perfect equilibrium, as we require that the strategies are a NE after every possible deviation, although the player might not have complete information as to what the deviation is.

Intuitively, a profile $\vec{M}$ of TMs is a computational subgame-perfect equilibrium if for all players i, all histories h where i moves, and all memory profiles $\vec{m}$ of the players, there is no polynomial-time TM $\bar{M}$ such that player i can gain more than ϵ by switching from M_i to $\bar{M}$. To make it precise, we must again consider an infinite sequence of games of increasing size (just as we do for NE, although this definition is more complicated since we must consider memory states).

For a memory state m and a TM M let $M(m)$, stand for running M with ini-

tial memory state m. We use $\vec{M}(\vec{m})$ to denote $(M_1(m_1), \ldots, M_c(m_c))$. Let $p_i^{G,\delta}(\vec{M})$ denote player i's payoff in $G^\infty(\delta)$ when $\vec{M}$ is played.

Definition 5.4.1 *An infinite sequence of strategy profiles $\vec{M}^1, \vec{M}^2, \ldots$, where $\vec{M}^k = (M_1^k, ..., M_c^k)$, is a* computational subgame-perfect ϵ-equilibrium *of an infinite sequence of repeated games $G_1^\infty, G_2^\infty, \ldots$ where the size of G_k is k, if, for all players $i \in [c]$, all sequences $h_1 \in H_{G_1^\infty}, h_2 \in H_{G_2^\infty}, \ldots$ of histories, all sequences $\vec{m}^1, \vec{m}^2, \ldots$ of polynomial-length memory-state profiles, where $\vec{m}^k = (m_1^k, \ldots, m_c^k)$, and all non-uniform PPT adversaries $\bar{M}$, there exists k_0 such that, for all $k \geq k_0$,*

$$p_i^{G_k^\infty(h_k),\delta}(\bar{M}(m_i^k), \vec{M}_{-i}^k(\vec{m}_{-i}^k)) \leq p_i^{G_k^\infty(h_k),\delta}(\vec{M}^k(\vec{m}^k)) + \epsilon(k).$$

5.4.2 Computing a subgame-perfect ϵ-NE

For a game G such that $|G| = n$, and a polynomial ℓ, consider the following strategy $\sigma^{NE,\ell}$, and let $\vec{M}^{\sigma^{NE,\ell}}$ be the TMs that implement this strategy. This strategy is similar in spirit to that proposed in Section 5.3.2; indeed, the first two phases are identical. The key difference is that the punishment phase is played for only $\ell(n)$ rounds. After that, players return to phase 1. As we show, this limited punishment is effective since it is not played long enough to make it an empty threat (if ℓ is chosen appropriately). Phase 4 takes care of one minor issue: The fact that we can start in any memory state means that a player might be called on to do something that, in fact, he cannot do (because he doesn't have the information required to do it). For example, he might be called upon to play the correlated punishment strategy in a state where he has forgotten the random seed, so he cannot play it. In this case, a default action is played. Note that his was not an issue in the analysis of NE.

1. Play according to sq (with wraparound) as long as all players played according to sq in the previous round.

2. After detecting a deviation by player $j \neq i$ in round t_0:[10]

   (a) Generate a pair (pk_i, sk_i) using Gen(1^n). Store sk_i in memory and use the next $v(n)$ rounds to broadcast pk_i.

   (b) If $i = j + 1$ (with wraparound), player i does the following:

   - i records $pk_{j'}$ for all players $j' \notin \{i, j\}$;
   - i generates a random n-bit string $seed$;
   - for each player $j' \notin \{i, j\}$, i computes $m = Enc_{pk_{j'}}(seed)$, and uses the next $(c-2)z(n)$ rounds to communicate these strings to the players other than i and j (in some predefined order).

   (c) If $i \neq j + 1$, player i does the following:

   - i records the actions played by $j + 1$ at time slots designated for i to retrieve $Enc_{Pk_i}(seed)$;
   - i decrypts to obtain $seed$, using Dec and sk_i.

3. Phase 2 ends after $v(n) + (c-2)z(n)$ rounds. The players other than j then compute $PS_{seed}(t)$ and use it to determine which action profile to play according to the distribution defined by a fixed (correlated) punishment strategy against j. Player j plays his best response to the correlated punishment strategy throughout this phase. After $\ell(n)$ rounds, they return to phase 1, playing the sequence sq from the point at which the deviation occurred (which can easily be inferred from the history).

[10] Again, if more than one player deviates while playing sq, the players punish the one with the smaller index. The punished player plays his best response to what the other players are doing in this phase.

4. If at any point less than or equal to $v(n) + (c-2)z(n)$ time steps from the last deviation from phase 1 the situation is incompatible with phase 2 as described above (perhaps because further deviations have occurred), or at any point between $v(n)+(c-2)z(n)$ and $v(n)+(c-2)z(n)+\ell(n)$ steps since the last deviation from phase 1 the situation is incompatible with phase 3 as described above, play a fixed action for the number of rounds left to complete phases 2 and 3 (i.e., up to $v(n) + (c-2)z(n) + \ell(n)$ steps from the last deviation from phase 1). Then return to phase 1.

Note that with this strategy a deviation made during the punishment phase is not punished. Phase 2 and 3 are always played to their full length (which is fixed and predefined by ℓ and z). We say that a history h is a phase 1 history if it is a history where an honest player should play according to sq. History h is a phase 2 history if it is a history where at most $v(n) + (c-2)z(n)$ rounds have passed since the last deviation from phase 1; h is a phase 3 history if more than $v(n) + (c-2)z(n)$ but at most $v(n) + (c-2)z(n) + \ell(n)$ rounds have passed since the last deviation from phase 1. No matter what happens in phase 2 and 3, a history in which exactly $v(n) + (c-2)z(n) + \ell(n)$ round have passed since the last deviation from phase 1 is also a phase 1 history (even if the players deviate from phase 2 and 3 in arbitrary ways). Thus, no matter how many deviations occur, we can uniquely identify the phase of each round.

We next show that by selecting the right parameters, these strategies are easy to compute and are a subgame-perfect ϵ-equilibrium for all inverse polynomials ϵ.

Definition 5.4.2 *Let $\mathcal{G}_{a,b,c,n}$ be the set of all games with c players, at most n actions per player, integral payoffs, maximum payoff a, and minimum payoff b.*

We first show that for any strategy that deviates while phase 1 is played, there is a strategy whose payoff is at least as good and either does not deviate in the first polynomially many rounds, or after its first deviation, deviates every time phase 1 is played. (Recall that after every deviation in phase 1, the other players play the punishment phase for $\ell(n)$ rounds and then play phase 1 again.)

We do this by showing that if player i has a profitable deviation at some round t of phase 1, then it must be the case that every time this round of phase 1 is played, i has a profitable deviation there. (That is, the strategy of deviating every time this round of phase 1 is played is at least as good as a strategy where player i correlates his plays in different instantiations of phase 1.) While this is trivial in traditional game-theoretic analyses, naively applying it in the computational setting does not necessarily work. It requires us to formally show how we reduce a polynomial time TM M to a different TM M' of the desired form without blowing up the running time and size of the TM.

For a game G, let $H^{1,n,f}_{G^\infty}$ be the set of histories h of G^∞ of length at most $nf(n)$ such that at (the last node of) h, $\sigma^{NE,\ell}$ is in phase 1. Let $R(M)$ be the polynomial that bounds the running time of TM M.

Definition 5.4.3 *Given a game G, a deterministic TM M is said to be* (G, f, n)-well-behaved *if, when $(M, \sigma^{NE,\ell}_{-i})$ is played, then either M does not deviate for the first $nf(n)$ rounds or, after M first deviates, M continues to deviate from sq every time phase 1 is played in the next $nf(n)$ rounds.*

Lemma 5.4.4 *For all a, b, c, and all polynomials f, there exists a polynomial g such that for all n, all games $G \in \mathcal{G}_{a,b,c,n}$, all $h \in H^{1,n,f}_{G^\infty}$, all players i, and all TMs M, there exists a $(G(h), f, n)$-well-behaved TM M' such that $p_i^{G^h,1/f(n)}(M', \vec{M}^{\sigma^{NE,\ell}}_{-i}) \geq$*

$p_i^{G^h,1/f(n)}(M, \vec{M}_{-i}^{\sigma^{NE,\ell}})$, *and* $R(M'), |M'| \leq g(R(M))$.

Proof: Suppose that we are given $G \in \mathcal{G}_{a,b,c,n}$, $h \in H_{G^\infty}^{1,n,f}$, and a TM M. We can assume without loss of generality that M is deterministic (we can always just use the best random tape). If M does not deviate in the first $nf(n)$ rounds of $G(h)^\infty$ then M' is just M, and we are done. Otherwise, we construct a sequence of TMs starting with M that are, in a precise sense, more and more well behaved, until eventually we get the desired TM M'.

For $t_1 < t_2$, say that M is (t_1, t_2)-(G, f, n)-*well-behaved* if M does not deviate from sq until round t_1, and then deviates from sq every time phase 1 is played up to (but not including) round t_2 (by which we mean there exists some history in which M does not deviate at round t_2 and this is the shortest such history over all possible random tapes of $\vec{M}_{-i}^{\sigma^{NE,\ell}}$). We construct a sequence $M_1, M_2, \ldots$ of TMs such that (a) $M_1 = M$, (b) M_i is (t_1^i, t_2^i)-(G, f, n)-well-behaved, (c) either $t_1^{i+1} > t_i$ or $t_1^{i+1} = t_1^i$ and $t_2^{i+1} > t_2^i$, and (d) $p_i^{G^h,1/f(n)}(M_{i+1}, \vec{M}_{-i}^{\sigma^{NE,\ell}}) \geq p_i^{G^h,1/f(n)}(M_i, \vec{M}_{-i}^{\sigma^{NE,\ell}})$. Note that if $t_1 \geq nf(n)$ or $t_2 \geq t_1 + nf(n)$, then a (t_1, t_2)-(G, f, n)-well-behaved TM is (G, f, n)-well-behaved.

Let $t < nf(n)$ be the first round at which M deviates. (This is well defined since the play up to t is deterministic.) Let the history up to time t be h^t. If M deviates every time that phase 1 is played for the $nf(n)$ rounds after round t, then again we can take $M' = M$, and we are done. If not, let t' be the first round after t at which phase 1 is played and there exists some history of length t' at which M does not deviate. By definition, M is (t, t')-(G, f, n)-well behaved. We take $M_1 = M$ and $(t_1^1, t_2^1) = (t, t')$. (Note that since $\vec{M}_{-i}^{\sigma^{NE,\ell}}$ are randomized during phase 2, the first time after t at which M returns to playing phase 1 and does not deviate may depend on the results of their coin tosses. We take t' to be

the first time this happens with positive probability.)

Let s^{h^*} be M's memory state at a history h^*. We assume for ease of exposition that M encodes the history in its memory state. (This can be done, since the memory state at time t is of size polynomial in t.) Consider the TM M'' that acts like M up to round t, and copies M's memory state at that round (i.e., s^{h^t}). M'' continues to plays like M up to the first round t' with $t < t' < t + nf(n)$ at which $\sigma^{NE,\ell}$ would be about to return to phase 1 and M does not deviate (which means that M plays an action in the sequence sq at round t'). At round t', M'' sets its state to s^{h^t} and simulates M from history h^t with states $s^{h(t)}$; so, in particular, M'' does deviate at time t'. (Again, the time t' may depend on random choices made by $\vec{M}_{-i}^{\sigma^{NE,\ell}}$. We assume that M'' deviates the first time M is about to play phase 1 after round t and does not deviate, no matter what the outcome of the coin tosses.) This means, in particular, that M'' deviates at any such t'. We call M'' a *type 1 deviation from* M.

If $p_i^{G^{h^t},1/f(n)}(M'', \vec{M}_{-i}^{\sigma^{NE,\ell}}) > p_i^{G^{h^t},1/f(n)}(M, \vec{M}_{-i}^{\sigma^{NE,\ell}})$, then we take $M_2 = M''$. Note that $t_1^2 = t_1^1 = t$, while $t_2^2 > t_2^1 = t'$, since M'' deviates at t'. If $p_i^{G^{h^t},1/f(n)}(M'', \vec{M}_{-i}^{\sigma^{NE,\ell}}) < p_i^{G^{h^t},1/f(n)}(M, \vec{M}_{-i}^{\sigma^{NE,\ell}})$, then there exists some history h^* of both M and M'' such that $t < |h^*| < t + nf(n)$, M'' deviates at h^*, M does not, and M has a better expected payoff than M'' at h^*. (This is a history where the type 1 deviation failed to improve the payoff.) Take M_2 to be the TM that plays like $\vec{M}_i^{\sigma^{NE,\ell}}$ up to time t, then sets its state to s^{h^*}, and then plays like M with state s^{h^*} in history h^*. We call M_2 a *type 2 deviation from* M. Note that M_2 does not deviate at h^t (since M did not deviate at history h^*). Let $\delta' = (1-\delta)^{|h^*|-|h^t|}$. Clearly $\delta' p_i^{G^{h^t},1/f(n)}(M_2, \vec{M}_{-i}^{\sigma^{NE,\ell}}) = p_i^{G^{h^*},1/f(n)}(M, \vec{M}_{-i}^{\sigma^{NE,\ell}})$, since $\vec{M}_{-i}^{\sigma^{NE,\ell}}$ acts the same in G^{h^t} and G^{h^*}. Since M'' plays like $M(s^{h_t})$ at h^*,

$p_i^{G^{h^*},1/f(n)}(M'', \vec{M}_{-i}^{\sigma^{NE,\ell}}) = \delta' p_i^{G^{h^t},1/f(n)}(M, \vec{M}_{-i}^{\sigma^{NE,\ell}})$. Combining this with the previous observations, we get that $p_i^{G^{h^t},1/f(n)}(M_2, \vec{M}_{-i}^{\sigma^{NE,\ell}}) \geq p_i^{G^{h^t},1/f(n)}(M, \vec{M}_{-i}^{\sigma^{NE,\ell}})$. Also note that $t_1^2 > t_1^1$. This completes the construction of M_2. We inductively construct M_{i+1}, $i = 2, 3, \ldots$, just as we did M_2, letting M_i play the role of M.

Next observe that, without loss of generality, we can assume that this sequence arises from a sequence of type 2 deviations, followed by a sequence of type 1 deviations: For let j_1 be the first point in the sequence at which a type 1 deviation is made. We claim that we can assume without loss of generality that all further deviations are type 1 deviations. By assumption, since M_{j_1} gives i higher utility than M_{j_1-1}, it is better to deviate the first time M_{j_1-1} wants to play phase 1 again after an initial deviation. This means that when M_{j_1} wants to play phase 1 again after an initial deviation it must be better to deviate again, since the future play of the $\vec{M}_{-i}^{\sigma^{NE,\ell}}$ is the same in both of these situations. This means that once a type 1 deviation occurs, we can assume that all further deviations are type 1 deviations.

Let M_j be the first TM in the sequence that is well behaved. (As we observed earlier, there must be such a TM.) Using the fact that the sequence consists of a sequence of type 2 deviations followed by a sequence of type 1 deviations, it is not hard to show that M_j can be implemented efficiently. First notice that M_{j_1} is a TM that plays like $\vec{M}_i^{\sigma^{NE,\ell}}$ until some round, and then plays M starting with its state at a history which is at most $(nf(n))^2$ longer than the real history at this point. This is because its initial history becomes longer by at most $nf(n)$ at each round and we iterate this construction at most $nf(n)$ times. This means that its running time is obviously polynomially related to the running time of the original M. The same is true of the size of M_{j_1}, since we need to encode

only the state at this initial history and the history at which we switch, which is polynomially related to $R(M)(n)$.

To construct M_j, we need to modify M_{j_1} only slightly, since only type 1 deviations occur. Specifically, we need to know only $t^1_{j_1}$ and to encode its state at this round. At every history after that, we run M_{J_1} (which is essentially running M on a longer history) on a fixed history, with a potential additional step of copying the state. It is easy to see that the resulting TM has running time and size at most $O(R(M))$. □

We now state and prove our theorem, which shows that there exists a polynomial-time algorithm for computing a subgame-perfect ϵ-equilibrium by showing that, for all inverse polynomials ϵ, there exists a polynomial function ℓ of ϵ such that $\sigma^{NE^*,\ell}$ is a subgame-perfect ϵ-equilibrium of the game. The main idea of the proof is to show that the players can't gain much from deviating while the sequence is being played, and also that, since the punishment is relatively short, deviating while a player is being punished is also not very profitable.

Theorem 5.4.5 *For all a, b, c, and all polynomials q, there is a polynomial f and a polynomial-time algorithm F such that, for all sequences $G_1, G_2, \ldots$ of games with $G^j \in G_{a,b,c,j}$ and for all inverse polynomials $\delta \leq 1/f$, the sequence of outputs of F given the sequence $G_1, G_2, \ldots$ of inputs is a subgame-perfect $\frac{1}{q}$-equilibrium for $G_1^\infty(\delta(1)), G_2^\infty(\delta(2)), \ldots$.*

Proof: Given a game $G^n \in \mathcal{G}(a, b, c, n)$, the algorithm finds a correlated equilibrium σ of G^n, which can be done in polynomial time using linear programming. Each player's expected payoff is at least 0 when playing σ, since we assumed

that the minimax value of the game is 0. Let $r = a - b$. By Lemma 5.3.3 and Lemma 5.3.4, we can construct a sequence sq of length $w(n) = 4(rnq(n) + 1)n^c$ and set $f'(n) = 4rw(n)q(n)$, so that if the players play sq infinitely often and $\delta < 1/f'(n)$, then all the players get at least $-1/2q(n)$. The correlated punishment strategy against each player can also be found in polynomial time using linear programming.

Let $m(n) = v(n) + (c-2)z(n) + 1$ (the length of phase (2) plus the round of deviation). Let $\ell(n) = nq(n)(m(n)a + 1)$, let σ_n^* be the strategy $\vec{M}^{\sigma^{NE},\ell}$ described above, and let $f(n) = \max(3rq(n)(\ell(n) + m(n)), f'(n))$.

We now show that $\sigma_1^*, \sigma_2^*, \ldots$ is a subgame-perfect $(1/q)$-equilibrium for every inverse polynomial discount factor $\delta \leq 1/f$. We focus on deviations at histories of length $< \frac{n}{\delta(n)}$, since, by Lemma 5.3.5, the sum of payoffs received after that is negligible. Thus, there exists some n_0 such that, for all $n > n_0$, the payoff achieved after that history is less than $1/q(n)$, which does not justify deviating.

We first show that no player has an incentive to deviate in subgames starting from phase 1 histories. By Lemma 5.4.4, it suffices to consider only a deviating strategy that after its first deviation deviates every time phase 1 is played; for every deviating strategy, either not deviating does at least as well or there is a deviating strategy of this form that does at least as well. Let h_1 be the history in which the deviation occurs and let M be the deviating strategy. Notice that $\vec{M}^{\sigma^{NE},\ell}$ can always act as intended at such histories; it can detect it is in such a history and can use the history to compute the next move (i.e., it does not need to maintain memory to figure out what to do next).

The player's payoff from $(M, \vec{M}_{-i}^{\sigma^{NE,\ell}})$ during one cycle of deviation and punishment can be at most a at each round of phase 2 and, by Lemma 5.3.6, is negligible throughout phase 3. (We use ϵ_{neg} to denote the negligible payoff to a deviator in phase 3.) Thus, the payoff of the deviating player from $(M, \vec{M}_{-i}^{\sigma^{NE,\ell}})$ from the point of deviation onwards is at most

$$((1-\delta(n)^{|h_1|})\big(\delta(n)(m(n)a+\epsilon_{neg})\sum_{t=0}^{\lceil\frac{nf(n)-|h_1|}{m(n)+\ell(n)}\rceil}(1-\delta(n))^{(m(n)+\ell(n))t}+\epsilon'_{neg}\big)$$
$$\leq ((1-\delta(n)^{|h_1|})\big(\delta(n)(m(n)a+\epsilon_{neg})\sum_{t=0}^{\infty}(1-\delta(n))^{(m(n)+\ell(n))t}+\epsilon'_{neg}\big),$$

where ϵ'_{neg} is the expected payoff after round $nf(n)$. By Lemma 5.3.3, no matter where in the sequence the players are, the average discounted payoff at that point from playing honestly is at least $-1/2q(n)$. Thus, the payoff from playing $(\vec{M}^{\sigma^{NE,\ell}})$ from this point onwards is at least $-(1-\delta(n))^{|h_1|})1/2q(n)$. We can ignore any payoff before the deviation since it is the same whether or not the player deviates, and also divide both sides by $(1-\delta(n))^{|h_1|})$; thus, it suffices to prove that

$$\delta(n)(m(n)a+\epsilon_{neg})\sum_{t=0}^{\infty}(1-\delta(n))^{(m(n)+\ell(n))t}+\epsilon'_{neg}\leq\frac{1}{q(n)}-\frac{1}{2q(n)}.$$

The term on the left side is bounded by $O\big(\frac{m(n)a+\epsilon_{neg}}{nq(n)(m(n)a+1)}\big)$, and thus there exists n_1 such that, for all $n > n_1$, the term on the left side is smaller than $\frac{1}{2q(n)}$ (In fact, for all constants c, there exists n_c such that the left-hand side is at most $\frac{1}{cq(n)}$ for any $n > n_c$.)

We next show that no player wants to deviate in phase 2 or 3 histories. Notice that since these phases are carried out to completion even if the players deviate while in these phases (we do not punish them for that), the honest strategy can easily detect whether it is in such a phase by looking at when the last deviation

from phase 1 occurred. First consider a punishing player. By not following the strategy, he can gain at most r for at most $\ell(n) + m(n)$ rounds over the payoff he gets with the original strategy (this is true even if his memory state is such that he just plays a fixed action, or even if another player deviates while the phase is played). Once the players start playing phase 1 again, our previous claim shows that no matter what the actual history is at that point, a strategy that does not follow the sequence does not gain much. It is easy to verify that, given the discount factor, a deviation can increase his discounted payoff by at most $\frac{1}{q(n)}$ in this case. (Notice that the previous claim works for any constant fraction of $1/q(n)$, which is what we are using here since the deviation in the punishment phase gains $1/cq(n)$ for some c.)

The punished player can deviate to a TM that correctly guessed the keys chosen (or the current TM's memory state might contain the actual keys and he defects to a TM that uses these keys), in which case he would know exactly what the players are going to do while they are punishing him. Such a deviation exists once the keys have been played and are part of the history. Another deviation might be a result of the other TMs being in an inconsistent memory state, so that they play a fixed action, one which the punished player might be able to take advantage of. However, these deviations work (or any other possible deviation) only for the current punishment phase. Once the players go back to playing phase 1, this player can not gain much by deviating from the sequence again. For if he deviates again, the other players will choose new random keys and a new random seed (and will have a consistent memory state); from our previous claims, this means that no strategy can gain much over a strategy that follows the sequence. Moreover, he can also gain at most r for at most $\ell(n) + m(n)$ rounds which, as claimed before, means that his discounted payoff difference is

less than $\frac{1}{q(n)}$ in this case.

This shows that, for n sufficiently large, no player can gain more than $1/q(n)$ from deviating at any history. Thus, this strategy is a subgame-perfect $1/q$-equilibrium. □

Using the same arguments as in Section 5.3.3, we can also apply these ideas to efficiently find a computational subgame-perfect ϵ-equilibrium in constant-degree graphical games.

Corollary 5.4.6 *For all a, b, d, and all polynomials q, there is a polynomial f and a polynomial-time algorithm F such that, for all sequences $G_1, G_2, \ldots$ of games with $G^j \in G_{a,b,d,j,j}$ and for all inverse polynomials $\delta \leq 1/f$, the sequence of outputs of F given the sequence $G_1, G_2, \ldots$ of inputs is a subgame-perfect $\frac{1}{q}$-equilibrium for $G_1^\infty(\delta(1)), G_2^\infty(\delta(2)), \ldots$.*

CHAPTER 6

COMPUTATIONAL EXTENSIVE-FORM GAMES

6.1 Introduction

In the previous section we studied infinitely repeated games, in which the input to the players naturally grows as part of the game, and thus it makes sense to talk about polynomial-time bounded agent. But how should we model cases where the game itself does not grow. Such games have been considered before in works [14, 26, 39, 40, 67] on solving game-theoretic problems using computationally bounded players. However there has not really been a careful study of these models and the solution concepts appropriate for such models. What does it mean, for example, to say that a fixed finite game played by polynomial-time players has a Nash equilibrium?

Consider for example the following two-player extensive-form game G given in Figure 6.1: At the empty history, player 1 secretly chooses one of two alternatives and puts her choice inside a sealed envelope. Player 2 then also chooses one of these two alternatives. Since player 2 acts without knowing 1's choice, the two histories where 1 made different choices are in the same information set of player 2. Finally, player 1 can either open the envelope and reveal her choice or destroy the envelope. If she opens the envelope and she chose a different alternative than player 2, player 1 wins and gets a utility of 1; otherwise (i.e., if player 1 either chose the same alternative as player 2 or she destroyed the envelope) player 1 loses and gets a utility of -1. Player's 2's utility is the opposite of player 1's.

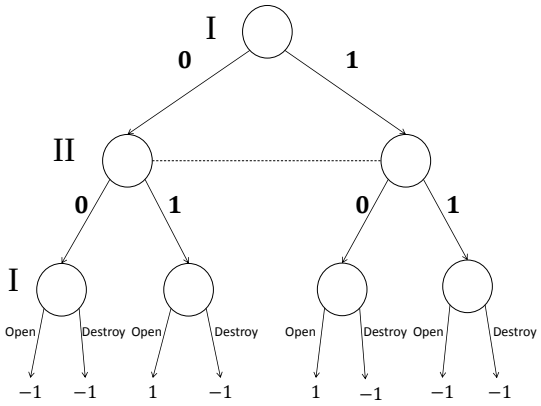

Figure 6.1: A simple coin tossing game.

Resource-bounded players can implement this game even without access to envelopes, using what is called a *commitment scheme*. A commitment scheme is a two-phase two-party protocol involving a sender (player 1 above) and a receiver (player 2). The sender sends the receiver a message in the first phase that commits him to a bit without giving the receiver information about the bit (at least no information that he can efficiently compute from the message); this is the computational analogue of putting the bit in an envelope. In the second phase, the sender "opens the envelope" by sending the receiver some information that allows the receiver to confirm what bit the sender committed to in the first phase. Thus, we can talk about a game $\mathcal{G}$ (actually a sequence of games as discussed later) where instead of player 1 using an abstract envelope to send her choice to player 2, she uses a commitment scheme to do so. See Figure 6.2 for an example of such a game.

Intuitively, we would like to say that the two games represent the same underlying game. However, there are many subtleties in doing so. To get a sense of the problems, note that to use commitment schemes we need the players to be computationally bounded. But to talk about computation bounds (for instance,

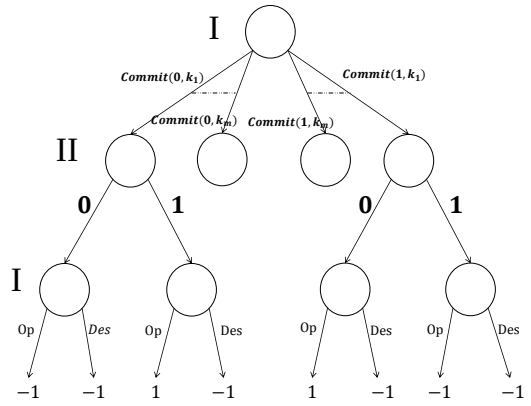

Figure 6.2: A coin tossing game with commitments.

polynomial-time TMs), we need to have a sequence of inputs that can grow as a function of n. So how do we proceed if we want to talk about computationally bounded players in a fixed finite game? The idea is that we will have a sequence of games, potentially increasing in size, that represents the single game. As we shall see, the information structure of the games in the infinite sequence might differ from that of the underlying game. For example, in the games described before, while a commitment scheme gives no information to a computationally bounded player, an unbounded player has complete information; the encrypted string uniquely identifies the bit that was committed. Thus, unlike in G, commitments to different bits in $\mathcal{G}$ are in different information sets for player 2.

Additional complications arise when we consider solution concepts for such games. Traditional notions of equilibrium involve all players making a best response. But if we restrict to computationally bounded players, there may not be a best response, especially for the kinds of cryptographic problems that we would like to consider. For example, for every polynomial-time TM, there may be another TM that does a little better by spending a little longer trying to do decryption. (See [29] for an example of this phenomenon.) Moreover, when con-

sidering sequentially rational solution concepts it is unclear what information structure should be considered since, as we discussed, the information structure of the computational games does not capture the knowledge of computationally bounded players.

Our contributions. As a first step to capturing these notions, in Section 6.3.1, we define what it means for a sequence $\mathcal{G} = (G_1, G_2, \ldots)$ of games to represent a single game G. Intuitively, all the games in the sequence $\mathcal{G}$ have the same basic structure as G, but might use increasingly longer strings to represent actions in G (e.g., an action a in G might be represented in G_n by an encryption of a that uses a security parameter of length n). More precisely, we require a mapping from histories in the games G_n to histories in G, as well as a mapping from strategies in G to strategies in $\mathcal{G}$, and impose what we argue are reasonable conditions on these mappings.[1] In Section 6.3.2, we show how this definition play out in the example discussed above.

As hinted before, our conditions do not force the games in $\mathcal{G}$ to have the same information structure as G. While two histories in the same information set in G_n must map to two histories in the same information set in G, it may also be the case that two histories in different information sets in G_n are mapped to the same information set in G. Although a player can distinguish two histories in different information sets (for example a commitment to 0 and a commitment to 1 in the example are two different strings), at a computational level, she cannot tell them apart. The encodings just look like random strings to her. There

[1]The idea of games that depends on a security parameter goes back to Dodis, Halevi, and Rabin [14]. Hubáček and Park [40] also consider a mapping between histories in a computational game and histories in an abstract game, although they do not consider the questions in the same generality that we do here.

is a sense in which she, as a computationally bounded player, does not understand the "meaning" of these histories (although a computationally unbounded player could break the commitment and tell them apart). In Section 6.3.3, we make this intuition precise, showing that our requirements force all histories that map to the same information set in G to be *computationally indistinguishable*, even if they are in different information sets in $\mathcal{G}$.

Once we have defined our model of computational games, we focus on defining analogues of two solution concepts, Nash equilibrium (NE) and sequential equilibrium. In Section 6.4.1, we define a computational analogues of NE, which considers only deviations that can be implemented by polynomial-time TMs. It handles previously mentioned complications by allowing for the strategy to be an ϵ best response for some negligible function ϵ. (Our definition of NE is similar in spirit to the definition in Dodis, Halevi, and Rabin [14].) We show that if a strategy profile is a NE in the underlying game G, then there is a corresponding strategy profile of polynomial time TMs that is a computational NE in $\mathcal{G}$. Thus, we provide conditions that guarantee the existence of a computational NE, addressing an open question of Katz [45].

In Section 6.4.2, we define a computational analogue of sequential equilibrium. It is notoriously problematic to define sequentially rational solution concepts in cryptographic protocols. For example, Gradwohl, Livne, and Rosen [26] provide a general discussion of the issue, and give a partial solution in terms of avoiding what they call "empty threats", which applies only to two-player games of perfect information, and discuss possible extensions. Our notion of computational sequential equilibrium, which is quite different in spirit from the solution concept of Gradwohl, Livne, and Rosen (and arguably conceptually

much simpler and much closer in spirit to the standard game-theoretic definition), applies to arbitrary sequence of games that represent a finite game, and uses the intuitions we develop on the connection between information sets in the underlying finite game and computational indistinguishability in the sequence. We again show that if a strategy profile is a sequential equilibrium in the underlying game G, then there is a corresponding strategy profile of polynomial time TMs that is a computational sequential equilibrium in $\mathcal{G}$.

An important benefit of our approach is that it separates the game-theoretic analysis from the cryptographic analysis. We can view the sequence $\mathcal{G}$ as an implementation of an abstract game G. Given this view, we can first prove that a protocol is a good implementation of an abstract game, and then analyze the strategic aspects in that simple abstract game. For example, to show a prescribed cryptographic protocol is a Nash (resp., sequential) equilibrium, we can first show it represents an abstract ideal game; it then suffices to show that the protocol corresponds to a strategy profile that is a Nash (resp., sequential) equilibrium in the much simpler underlying game. We give an example of this idea in Section 6.5, where we show how our approach can be used to analyze a protocol for implementing a correlated equilibrium (CE) without a mediator using cryptography, in the spirit of the work of Dodis, Halevy, and Rabin [14].

6.2 Preliminaries

6.2.1 Extensive-form games

We begin by reviewing the formal definition of an extensive-form game (adapted from [56]). A finite extensive-form game G is a tuple $([c], H, P, \vec{u}, \vec{\mathcal{I}})$ satisfying the following conditions:

- $[c] = \{1, \ldots, c\}$ is the set of players in the game.
- H is a set of history sequences that satisfies the following two properties:
  - the empty sequence $\langle\,\rangle$ is a member of H;
  - if $\langle a_1, \ldots, a_K\rangle \in H$ and $L < K$ then $\langle a_1, \ldots, a_L\rangle \in H$. The elements of a history h are called *actions*.

  A history $\langle a_1, \ldots, a^K\rangle \in H$ is *terminal* if there is no a such that $\langle a^1, \ldots, a^K, a\rangle \in H$. The set of actions available after a nonterminal history h is denoted $A(h) = \{a : h \cdot a) \in H\}$ (where $h \cdots a$ is the result of concatenating a to the end of h.[2] Let H^T denote the set of terminal histories, let H^{NT} denote $H \setminus H^T$, and let H^i denote the histories after which player i plays.
- $P : H \setminus H^T \to [c]$. $P(h)$ specifies the player that moves at history h.
- $\vec{u} : H_T \to \mathbb{R}^c$ specifies for each terminal history the utility of the players at that history ($u_i(h)$ is the utility of player i at terminal history h).

[2]For technical convenience, we assume that $|A(h)| \geq 2$ for all histories h. If this is not the case, then that step of the game is not interesting, and can essentially be removed.

- for each player $i \in [c]$, $\mathcal{I}_i$ is a partition of H^i with the property that $A(h) = A(h')$ whenever h and h' are in the same member of the partition. For $I \in \mathcal{I}_i$, we denote by $A(I)$ the set $A(h)$ for $h \in I$ (recall that $A(h) = A(h')$ if h and h' are two histories in I). We assume without loss of generality that if $I \neq I'$, then $A(I)$ and $A(I')$ are disjoint (we can always rename actions to ensure that this is the case). We call $\mathcal{I}_i$ the *information partition* of player i; a set $I \in \mathcal{I}_i$ is an *information set* of player i; $\vec{\mathcal{I}} = (\mathcal{I}_1, \ldots, \mathcal{I}_c)$ is the *information partition structure* of the game. A game of *perfect information* is one where all the information sets are singletons.

This model can capture situations in which players forget what they knew earlier. Roughly speaking, a game has *perfect recall* if the information structure is such that the players remember everything they knew in the past.

Definition 6.2.1 *Let $EXP_i(h)$ be the record of player i's* experience *in history h, that is, all the actions he plays and all the information sets he encounters in the history. A game has* perfect recall *if, for each player i, we have $EXP_i(h) = EXP_i(h')$ whenever the histories h and h' are in the same information set for player i.*

A deterministic strategy s for player i is a function from $\mathcal{I}_i$ to actions, where for $I \in \mathcal{I}_i$, we require that $s(I) \in A(I)$. We also consider mixed strategies which are probability distribution over deterministic strategies. A profile of strategies $\vec{\sigma} = \{\sigma_1, \ldots, \sigma_c\}$ induces a distribution denoted $\rho_{\vec{\sigma}}$ on terminal histories. We say that a strategy profile is completely mixed if $\rho_{\vec{\sigma}}$ assigns positive probability to every history $h \in H^T$. The expected value of player i given $\vec{\sigma}$ is then $\sum_{h \in H^T} \rho_{\vec{\sigma}}(h) u_i(h)$. NE is then defined just as in normal-form games with these values.

Just as in the case of repeated games, NE is not a robust solution concept for extensive-form games, as it allows for empty threats. As we mentioned in the previous section, a more robust solution concept that also deals with the information structure of the game is *sequential equilibrium* [48]. An equivalent definition to the one given in the previous section, that does not require beliefs and is more suitable for the settings in this section is given by the following theorem:

Theorem 6.2.2 *[48, Proposition 6] Let G be an extensive-form game with perfect recall. There exists a belief system μ such that $(\vec{\sigma}, \mu)$ is a sequential equilibrium of G iff there exists a sequence of completely mixed strategy profiles $\vec{\sigma}^1, \vec{\sigma}^2, \ldots$ converging to $\vec{\sigma}$ and a sequence $\delta_1, \delta_2, \ldots$ of nonnegative real numbers converging to 0 such that, for each player i and each information set I for player i, $\vec{\sigma}_i^n$ is a δ_n-best response to $\vec{\sigma}_{-i}^n$ conditional on having reached I.*

6.2.2 Commitment schemes

We now define a cryptographic commitment scheme that will be used in our examples. Informally, such a scheme is a two-phase two-party protocol for a sender and a receiver. In the first phase, the sender sends a message to the receiver that commits the sender to a bit without giving the receiver any information about that bit; in the second phase, the sender reveals the bit to which he committed in a way that guarantees that this really is the bit he committed to.

Definition 6.2.3 *A* secure commitment scheme with perfect bindings *is a pair of PPT algorithms C and R such that:*

- *C takes as input a security parameter 1^k, a bit b, and a bitstring r, and outputs $C(1^k, b, r), C_2(1^k, b, r)$, where $C_1(1^k, b, r)$, called the* commitment string, *is a k-bit string, and $C_2(1^k, b, r)$, called the* commitment key, *is a $(k-1)$-bit string. We use $C(1^k, b)$ to denote the output distribution of algorithm $C(1^k, b, r)$ when r is chosen uniformly at random.*
- *R is a deterministic algorithm that gets as input two strings c and s and outputs $o \in \{0, 1, f\}$.*
- *(Hiding) $\{C_1(1^k, 0)\}_{k\in\mathbb{N}}$ and $\{C_1(1^k, 1)\}_{k\in\mathbb{N}}$ are computationally indistinguishable.*
- *(Perfect binding) $R(C_1(1^k, b, r), (C_2(1^k, b, r)) = b$ for all k and r; moreover, if $s \neq C_2(1^k, b, r)$, then $R(C_1(1^k, b, r)), s) \notin \{0, 1\}$.*

Cryptographers typically assume that secure commitment schemes with perfect bindings exist. (Their existence would follow from the existence of *one-way permutations*; see [20] for further discussion and formal definitions.)

6.3 Computational Extensive-Form Games

6.3.1 Definitions

Statements of computational difficulty typically say that there is no (possibly randomized) polynomial-time algorithm for solving a problem. To make sense of this, we need to consider, not just one input, but a sequence of inputs, getting progressively larger. Similarly, to make sense of computational games, we cannot consider a single game, but rather must consider a sequence of games that

grow in size. The games in the sequence share the same basic structure. This means that, among other things, they involve the same set of players, playing in the same order, with corresponding utility functions. To make this precise, we first start with a more general notion, which we call a *computable uniform sequence of games*.

Definition 6.3.1 *A* computable uniform sequence $\mathcal{G} = \{G_1, G_2, \ldots\}$ of games *is a sequence that satisfies the following conditions:*

- *All the games in the sequence involve the same set of players.*
- *Let H_n be the set of histories in G_n. There exists a polynomial p such that, for all nonterminal histories $h \in H_n^{NT}$, $A(h) \subseteq \{0,1\}^{\leq p(n)}$.*[3] *In addition, there is a PPT algorithm that, on input 1^n and a history h, determines whether $h \in H_n$.*
- *There exists a polynomial-time computable function P' from $\bigcup_{n=1}^{\infty}(H_n^{NT})$ to $[c]$. The function P_n in game $G_n \in \mathcal{G}$ is then P' restricted to H_n^{NT}.*
- *For each player i, there exists a polynomial-time computable function $u_i : \bigcup_{n=1}^{\infty} H_n^T \rightarrow \mathbb{R}$ such that the utility function of player i in game G_n is u_i restricted to H_n^T.*

We sometimes call a computable uniform sequence of games a computational game.

Computable uniform sequences of games already suffice to allow us to talk about polynomial-time strategies. A strategy M for player i in a computable uniform sequence $\mathcal{G} = (G_1, G_2, \ldots)$ is a probabilistic TM that takes as input a pair $(1^n, v)$, where v is a view for player i in G_n (discussed below), and outputs

[3] $\{0,1\}^{\leq p(n)}$ denotes the language consisting of bitstrings of length at most $p(n)$.

an action in $A(I)$. We assume that the TMs are *stateful*; they have a tape on which the random bits used in previous rounds are recorded. The *view* of a stateful TM M for player i in G_n is a tuple (v_I, r), where v_I is the representation of information set I and r contains the randomness that has been used thus far (so is nondecreasing from round to round). This can be viewed as having perfect recall of randomness, as the TMs are not allowed to "forget" the randomness they used. It is considered part of their experience so far in the same way as the actions that they played and the information sets that they visited.[4]

We next define what it means for a uniform sequence $\mathcal{G} = (G_1, G_2, \ldots)$ of games to *represent* an underlying game G. To explain different aspects of this definition, it is useful to go back to the example in the introduction and discuss what it means for a sequence $\mathcal{G}$ to represent the game G in Figure 6.1. As discussed before, we can implement this game using a commitment scheme. The point is that now we get, not one game, but a sequence of games, one for each choice of security parameter. Rather than putting a bit b in an envelope, in G_n player 1 sends $C_1(1^n, b)$. More precisely, he sends $C_1(1^n, b, r)$, for a string r chosen chosen uniformly at random. To then open the envelope, player 1 can just send $C_2(1^n, b, r)$ and any other string to destroy it.

Roughly speaking, we want all the games in $\mathcal{G}$ to have the same "structure" as G. We formalize this by requiring a surjective mapping f_n from histories in each game G_n in the sequence to histories in G. Note that f_n is not, in general, one-to-one. There may be many histories in G_n representing a single history in G. This can already be seen in our example; each of the histories in G_n where

[4] This assumption is equivalent to allowing the TM to have an additional tape on which it can save an arbitrary state. For any TM M that does this, there is an equivalent TM M' that has no additional tape, but simply reconstructs M's state by simulating M's computation from scratch using its view. This suffices, for example, to reconstruct a secret key that was generated in the first round, so it can be used in later rounds.

player 1 sends $C_1(1^n, 1, r)$ get mapped to the history in G where player 1 puts 1 in an envelope. Moreover, although $C_1(1^n, 0, r)$ and $C_1(1^n, 1, r)$ get mapped to histories in the same information set in G, they are *not* in the same information set in G_n; an exponential-time player can break the encryption and tell that they correspond to different bits being put in the envelope. Thus, the mapping f_n does not completely preserve the information structure. We require that h and $f_n(h)$ have the same length (same number of actions). Of course, the utility associated with a terminal history h in G_n is the same as that associated with history $f_n(h)$ in G.

The first three conditions below capture the relatively straightforward structural requirements above. The final requirement imposes conditions on the players' strategies, and is somewhat more complicated. Informally, the fourth requirement is that there is a mapping $\mathcal{F}$ from strategies in G to strategies in $\mathcal{G}$, where $\mathcal{F}(\sigma)$ "corresponds" to σ in some appropriate sense. But what should "correspond" mean? Let $\vec{M}$ be a strategy profile for $\mathcal{G}$. For each game $G_n \in \mathcal{G}$, $\vec{M}$ induces a distribution denoted $\psi_{\vec{M}}^{G_n}$ on the terminal histories in G_n. By applying f_n, we can push this forward to a distribution $\phi_{\vec{M}}^{G_n}$ on the terminal histories in G. A mixed strategy profile $\vec{\sigma}$ in G also induces a distribution on the terminal histories in G, denoted $\rho_{\vec{\sigma}}$.

Definition 6.3.2 *A strategy profile* $\vec{\sigma}$ corresponds *to* $\vec{M}$ *if* $\{\phi_{\vec{M}}^{G_n}\}_{n\in\mathbb{N}}$ *is* statistically close *to* $\{\rho_{\vec{\sigma}}\}_{n\in\mathbb{N}}$*: that is, if* H^T *are the terminal histories of* G*, then there exists a negligible function* ϵ *such that, for all* n*,*

$$\sum_{h\in H^T} |Pr_{\phi_{\vec{M}}^{G_n}}[h] - Pr_{\rho_{\vec{\sigma}}}[h]| \leq \epsilon(n).$$

So one requirement we will have is that, for all strategy profiles $\vec{\sigma}$ in G, $\vec{\sigma}$ corresponds to $(\mathcal{F}(\sigma_1), \ldots, \mathcal{F}(\sigma_n))$, which we abbreviate as $\mathcal{F}(\vec{\sigma})$. In addition, we require that the strategy profile $\mathcal{F}(\vec{\sigma})$ "knows" which underlying action it plays. We formalize this by requiring that, for strategy σ in the underlying game, there is a TM M^σ that, given view v for player i in $\mathcal{G}$, outputs the action in G corresponding to the action played by $\mathcal{F}(\sigma)$ given view v.

Finally, we require a partial converse to the correspondence requirement. It is clearly too much to expect a full converse. $\mathcal{G}$ has a richer structure than G; it allows for more ways for the players to coordinate than G. So we cannot expect every strategy profile in $\mathcal{G}$ to correspond to a strategy profile in G. Thus, we require only that strategies in a rather restricted class of strategy profiles in $\mathcal{G}$ correspond to a strategy in G: namely, ones where we start with a strategy of the form $\mathcal{F}(\vec{\sigma})$ (which, by assumption, corresponds to $\vec{\sigma}$), and allow one player to deviate. We must also use a weaker notion of correspondence here. For example, in the game in Figure 6.1, even if we start with a strategy of the form $\mathcal{F}(\vec{\sigma})$, the deviating strategy M_1' could be such that player 1 commits to 0 in G_n for n even, and commits to 1 in G_n for n odd. The strategy profile $(M_1', \mathcal{F}(\sigma_2))$ does not correspond to any strategy profile in G. Thus, the notion of correspondence that we consider in this case is that if i plays M_i' rather than $\mathcal{F}(\sigma_i)$, then there exists a sequence $\sigma_1', \sigma_2', \ldots$ of strategies in G, rather than a single strategy σ', and require only that the sequence $\{\phi^{G_n}_{(M_i', \mathcal{F}(\vec{\sigma}_{-i}))}\}_{n \in \mathbb{N}}$ be computationally indistinguishable from $\{\rho_{\vec{\sigma}}\}_{n \in \mathbb{N}}$, rather than being statistically close.

Definition 6.3.3 *A computable uniform sequence $\mathcal{G} = \{G_1, G_2, \ldots\}$* represents *an underlying game G if the following conditions hold:*

UG1. G and every game in $\mathcal{G}$ involve the same set of players.

UG2. For each game $G_n \in \mathcal{G}$, there exists a surjective mapping f_n from the histories in G_n to the histories in G such that

- *(a) $|h| = |f_n(h)|$;*
- *(b) the same player moves in h and $f_n(h)$;*
- *(c) if h' is a subhistory of h, then $f_n(h')$ is a subhistory of $f_n(h)$;*
- *(d) if h and h' are in the same information set in G_n, then $f_n(h)$ and $f_n(h')$ are in the same information set in G;*
- *(e) for $h \in H$ (a history of G), let $LA(h)$ denote the last action played in h; if h and h' are in the same information set in G_n, then for any a such that $h||a \in H_n$, $LA(f_n(h||a)) = LA(f_n(h'||a))$ (where $||$ is the concatenation operator).*

UG3. If h is a terminal history of G_n, then the utility of each player i is the same in h and $f_n(h)$.

UG4. There is a mapping $\mathcal{F}$ from strategies in G to strategies in $\mathcal{G}$ such that

- *(a) for all strategy profiles $\vec{\sigma}$ in G, $\vec{\sigma}$ corresponds to $\mathcal{F}(\vec{\sigma}) = (\mathcal{F}(\sigma_1), \ldots, \mathcal{F}(\sigma_n))$;*
- *(b) for each strategy σ for player i in G, there exists a polynomial-time TM M^σ that, given as input 1^n and a view v for player i in G_n that is reachable when player i plays $\mathcal{F}(\sigma_i)$ in G_n, returns an action for player i such that $LA(f_n(\mathcal{F}(\sigma)(1^n, v, r_T))) = M^\sigma(1^n, v, r_T)$, where r_T is the random tape used (remember that the view contains the randomness used so far);*
- *(c) for all strategy profiles $\vec{\sigma}$ in G, all players i, and all polynomial-time strategies M_i' for player i in $\mathcal{G}$, there exists a sequence $\sigma_1', \sigma_2', \ldots$ of strategies for player i in G such that $\{\phi^{G_n}_{(M_i', \mathcal{F}(\vec{\sigma}_{-i}))}\}_n$ is computationally indistinguishable from $\{\rho^G_{(\sigma_n', \vec{\sigma}_{-i})}\}_n$.*

Definition 6.3.3 requires the existence of a sequence $\vec{f} = (f_1, f_2 \ldots)$ in UG2 and a function $\mathcal{F}$ in UG4. When we want to refer specifically to f and $\mathcal{F}$, we say that $\mathcal{G}$ $\langle \vec{f}, \mathcal{F} \rangle$-represents G.

Note that UG2 requires that if h and h' are in the same information set in G_n, then $f_n(h)$ and $f_n(h')$ must be in the same information set in G. This means that we can view f_n as a map from information sets to information sets. However, it does *not* require the converse. As discussed above, in $\mathcal{G}$, an exponential-time player may be able to make distinctions between histories that cannot be made of the corresponding histories in the underlying game. We would like to be able to say that a polynomial-time player cannot distinguish h and h' if $f_n(h)$ and $f_n(h')$ are in the same information set. As we show later, these conditions allow us to make such a claim.

Also note that since the game is finite, to show UG4(a) and UG4(b) hold, it is enough to prove they hold for deterministic strategies. Given a mapping $\mathcal{F}$ that satisfies UG4(a) and (b) for deterministic strategies, we can extend it to mixed strategies in the obvious way: since a mixed strategy is just a probability distribution over finitely many deterministic strategies, it can be implemented by a TM that plays that probability distribution up to negligible precision over the corresponding mapping of the deterministic strategies (such an approximating distribution can be easily constructed in polynomial time). It is obvious that UG4(a) still holds. UG4(b) holds since using v and r_T, we can reconstruct which deterministic strategy σ' in the support of σ was actually used to reach v, and then use the corresponding TM $M^{\sigma'}$.

6.3.2 The commitment game as a uniform computable sequence

We now consider how these definitions play out in the game G in Figure 6.1 and the sequence $\mathcal{G} = (G_1, G_2, \ldots)$ described above where player 1 uses a commitment scheme as an envelope.

Lemma 6.3.4 *$\mathcal{G}$ represents G.*

Proof: First, we show that $\mathcal{G}$ is a computable uniform sequence. All the games in the sequence involve exactly 2 players; the set of histories in G_n is a subset of $\{0,1\}^{\leq n}$, and it is easy to compute the next player to act; finally, the utility functions are polynomial-time computable by using the TM R of the commitment scheme.

Next we show that the sequence represents G. There is an obvious mapping from histories of the games in the sequence to histories of G: a commitment to 0 is mapped to 0, a commitment to 1 is mapped to 1, the action of player 2 is just mapped to the action in G, player 1 providing the right key is mapped to action "open", and player 1 providing a wrong key is mapped to "destroy". Finally, it is easy to verify that UG3 (the condition on utilities) holds.

To show that UG4 holds, we need to define a function $\mathcal{F}$. A strategy for player 2 in G can't depend on player 1's action, since player 2's information set contains both actions. Thus, a deterministic strategy σ_2 for player 2 in G just plays an action in $\{0,1\}$; the corresponding strategy $\mathcal{F}(\sigma_2)$ just plays the same string. UG4(b) holds trivially in this case. To define $\mathcal{F}(\sigma_1)$ for a strategy σ_1 for player 1, we need to show how to implement each action of player 1. To play

b at the empty history in G_n, 1 plays the commitment string $C_1(1^n, b, r)$, where r is the randomness used by player 1 in the computation (which is then saved as the TM's state). To play the action "open", it computes $k = C_2(1^n, b, r)$; to play "destroy", it plays $k \oplus 1$ (a string other than the right key). It is easy to see that UG4(b) holds for strategies of player 1. Moreover, it is easy to see that $\mathcal{F}(\vec{\sigma})$ corresponds to $\vec{\sigma}$, so UG4(a) holds. We extend $\mathcal{F}$ to mixed strategies as described above.

To see that UG4(c) holds, observe that a strategy for player 1 in G_n can clearly be mapped to a strategy in G: At the empty history player 1 has some distribution over commitments to 0 and commitments to 1. This clearly maps to a distribution over putting 0 and 1 in the envelope. At the other nodes where player 1 moves, G_n induces a distribution over correctly revealing the commitment or doing some other action; again, this clearly maps to a distribution over "open" and "destroy" in the obvious way. Since a strategy M_1' for player 1 in $\mathcal{G}$ induces, for all n, a strategy $M_{1,n}'$ for player 1 in G_n, we can associate a sequence $(\sigma_1', \sigma_2', \ldots)$ with M_1'. It is easy to check that, for all strategies σ_2 for player 2 in G, $\{\phi^{G_n}_{(M_i', \mathcal{F}(\sigma_2))}\}_n$ is computationally indistinguishable from $\{\rho^G_{(\sigma_n', \sigma_2)}\}_n$.

We similarly want to associate with each strategy for player 2 in $\mathcal{G}$ a sequence of strategies in G. This is a little more delicate, since the information structure in G_n is not the same as that in G. Given a strategy σ_1 for player 1 in G, and an arbitrary polynomial-time strategy M_2 for player 2 in $\mathcal{G}$, let $P_n(b)$ be the probability that M_2 plays b when $(\mathcal{F}(\sigma_1), M_2)$ is played in G_n. Let σ_n' be the strategy in G that plays according to the same distribution. We now claim that $\{\phi^{G_n}_{(\mathcal{F}(\sigma_1), M_2)}\}_n$ is indistinguishable from $\{\rho^G_{\sigma_1, \sigma_n'}\}_n$. Assume, by way of contradiction, that it is not. This can happen only if, for infinitely many n, M_2 plays 0 and 1 with

non-negligibly different probabilities, depending on whether it is faced with a commitment to 0 or a commitment to 1. But that means that, for infinitely many n, it can distinguish those two events with non-negligible probability. This contradicts the assumption that the commitment scheme is secure. □

6.3.3 Consistent partition structures

In this section, we discuss the connection between computational indistinguishability and information structure in games. As we saw, when going from the game G in Figure 6.1 to the game $\mathcal{G}$ that represents it, we replaced the information set in G (the use of an envelope) with computational indistinguishability (a commitment scheme). Although the games in $\mathcal{G}$ are perfect information games, so that the players have complete information about a history, if player 1 uses the commitment scheme appropriately, then player 2 does not really understand the "meaning" of a history (i.e., whether it represents a commitment to 0 or a commitment to 1). On the other hand, if player 1 "cheats" by using, for example, some low-entropy random string for the commitment, player 2 might have a strategy that is able to understand the "meaning" of its action. Thus, there is a sense in which the information structure of a computational game depends on the strategies of the players. This dependence on strategies does not exist in standard games. If each of two histories h and h' in some information set I for player i has a positive probability of being reached by a particular strategy profile, then when player i is in I, he will not know which of h or h' was played, even if he knows exactly what strategies are being played. The situation is different for computational games, in a way we now make precise.

Suppose that $\mathcal{G} = (G_1, G_2, \ldots)$ $\langle \vec{f}, \mathcal{F} \rangle$-represents G and h is a history of G, so that $f_n^{-1}(h)$ is the set of histories of G_n that are mapped to h by f_n. For a set H of histories of a game $G_n \in \mathcal{G}$, let $\mathcal{V}_n(H)$ be the set of views that a player can have at histories in H when G_n is played. For a strategy profile $\vec{M}$ in $\mathcal{G}$, let $\xi_{\vec{M}}^{G_n}(v)$ be the probability of reaching view $v \in \mathcal{V}_n(H)$ if the players play strategy profile $\vec{M}$ in G_n. For a set V of views, let $\xi_{\vec{M}}^{G_n}(V) = \sum_{v \in V} \xi_{\vec{M}}^{G_n}(v)$. For a set V of mutually incompatible views (i.e., a set V of views such that for all distinct views $v, v' \in V$, the probability of reaching v given that v' is reached is 0, and vice versa), let $X_{\vec{M},n}^{V}$ be a probability distribution on V such that $X_{\vec{M},n}^{V}(v) = \frac{\xi_{\vec{M}}^{G_n}(v)}{\xi_{\vec{M}}^{G_n}(V)}$ if $\xi_{\vec{M}}^{G_n}(V) > 0$, and $\frac{1}{|V|}$ otherwise. Let $\xi_{\vec{\sigma}}^{G}(S)$ denote the probability of reaching a set S of histories in G if the players play strategy profile $\vec{\sigma}$. Note that if $\xi_{\vec{\sigma}}^{G}(S) > 0$, then by UG4, for all sufficiently large n, we must have $\xi_{\vec{M}_{\vec{\sigma}}}^{G_n}(\mathcal{V}_n(f_n^{-1}(S))) > 0$.

Definition 6.3.5 *Let $\mathcal{G}$ $\langle \vec{f}, \mathcal{F} \rangle$-represent a game G and let $\vec{M}$ be a strategy in $\mathcal{G}$. A partition $\mathcal{I}_i$ of H^i (recall that this is the set of histories in G where i plays) is* $\vec{M}$-consistent *for player i if, for all non-singleton $I \in \mathcal{I}_i$ and all $h \in I$ such that both $\xi_{\vec{M}}^{G_n}(\mathcal{V}_n(f_n^{-1}(h)))$ and $\xi_{\vec{M}}^{G_n}(\mathcal{V}_n(f_n^{-1}(I \setminus h)))$ are non-negligible, $\{X_{\vec{M},n}^{\mathcal{V}_n(f_n^{-1}(h))}\}_{n \in \mathbb{N}}$ is computationally indistinguishable from $\{X_{\vec{M},n}^{\mathcal{V}_n(f_n^{-1}(I \setminus \{h\}))}\}_{n \in \mathbb{N}}$. A partition structure $\vec{I}$ is $\vec{M}$-consistent if, for all agents i, $\vec{\mathcal{I}}_i$ is $\vec{M}$-consistent.*

Intuitively, a partition $\mathcal{I}_i$ for player i is consistent with a strategy profile $\vec{M}$, if, when $\vec{M}$ is played in $\mathcal{G}$, for all $I \in \mathcal{I}_i$ and all histories $h, h' \in I$, the distribution over views that map to h is computationally indistinguishable from the distribution over views that map to h'. In our example, this means that player 2 can't distinguish between the distribution created by a commitment to 0 and the distribution created by a commitment to 1 if the commitment algorithm is run "honestly" (using truly random strings).

Note that we do not enforce any condition on histories in G that are mapped back to a set of histories that is reached with only negligible probability. This means there might be more than one $\vec{M}$-consistent information partition.

We next show that if $\mathcal{I}_i$ is the information partition of player i in G, and $\mathcal{G}$ $\langle \vec{f}, \mathcal{F} \rangle$-represent G then for any strategy profile $\vec{\sigma}$ in G, $\mathcal{I}_i$ must be $\mathcal{F}(\vec{\sigma})$-consistent.

Theorem 6.3.6 *If $\mathcal{G}$ $\langle \vec{f}, \mathcal{F} \rangle$-represents G, $\mathcal{I}_i$ is the information partition of player i in G, and $\vec{\sigma}$ is a strategy profile in G then $\mathcal{I}_i$ is $\mathcal{F}(\vec{\sigma})$-consistent.*

Proof: We must show that if $I \in \mathcal{I}_i$ is a non-singleton information set for i in G and $h \in I$, then for all strategy profiles $\vec{\sigma}$ in G such that $\xi^G_{\vec{\sigma}}(h) > 0$ and $\xi^G_{\vec{\sigma}}(I \setminus \{h\}) > 0$, $\{X^{\mathcal{V}_n(f_n^{-1}(h))}_{\mathcal{F}(\vec{\sigma}),n}\}_{n\in\mathbb{N}}$ is computationally indistinguishable from $\{X^{\mathcal{V}_n(f_n^{-1}(I\setminus\{h\}))}_{\mathcal{F}(\vec{\sigma}),n}\}_{n\in\mathbb{N}}$.

Assume, by way of contradiction, that $h \in I$, I is an information set for player i in G, and there exists a strategy profile $\vec{\sigma}$ in G that reaches both h and $I \setminus \{h\}$ with positive probability such that $\{X^{\mathcal{V}_n(f_n^{-1}(h))}_{\mathcal{F}(\vec{\sigma}),n}\}_n$ is distinguishable from $\{X^{\mathcal{V}_n(f_n^{-1}(I\setminus\{h\}))}_{\mathcal{F}(\vec{\sigma}),n}\}_n$. Thus, there exists a distinguisher D for these distributions. Let a and a' be distinct actions in $A(I)$. (Recall that we assumed that $|A(I)| \geq 2$.) Let M' be a strategy for player i in $\mathcal{G}$ such that when M' reaches a history that maps to I (by UG4(b) and the fact that the sets of actions available in each information set are disjoint, this can be checked in polynomial time), M' uses D to distinguish if its view is in $\mathcal{V}_n(f_n^{-1}(h))$ or $\mathcal{V}_n(f_n^{-1}(I \setminus \{h\}))$. M' then plays an action mapped to a if D returns 0 and an action mapped to a' otherwise. At a history other than one in $f_n^{-1}(I)$, M' plays like $\mathcal{F}(\sigma_i)$. It is easy to see that, because $\{X^{f_n^{-1}(h)}_{\mathcal{F}(\vec{\sigma}),n}\}_n$ and $\{X^{f_n^{-1}(I\setminus\{h\})}_{\mathcal{F}(\vec{\sigma}),n}\}_n$ are distinguishable with non-negligible

probability, there is a non-negligible probability that the strategy M' is able to detect which case holds, and play accordingly. That means that when histories of $(M', \mathcal{F}(\sigma_{-i}))$ are mapped to histories of G via f_n, there is a non-negligible gap between the probability of (h, a) and the probability of (h', a) for $h' \in I \setminus \{h\}$. Since $h \in I$, there can be no strategy σ' for player i such that (σ', σ_{-i}) has such a gap, and UG4(c) cannot hold. This gives us the desired contradiction. □

Note that Theorem 6.3.6 holds trivially if, for all $G_i \in \mathcal{G}$, all the histories of $\mathcal{G}$ that map to I are in the same information set in G_i. The theorem is of interest only when this is not the case. If we think of G as an abstract model of a computational game $\mathcal{G}$ that represents it, this result can be thought of as saying that information sets in G can model both real lack of information and computational indistinguishability in $\mathcal{G}$.

6.4 Solution Concepts for Computational Games

In this section, we consider analogues of two standard solution concepts in the context of computational games: Nash equilibrium and sequential equilibrium, and prove that they exist if the computational game represents a finite extensive-form game.

6.4.1 Computational Nash equilibrium

Informally, a strategy profile in $\mathcal{G}$ is a computational Nash equilibrium if no player i has a profitable *polynomial-time* deviation, where a deviation is taken to be profitable if it is profitable in infinitely many games in the sequence. Recall

that $\psi_{\vec{M}}^{G_n}$ is the distribution on the terminal histories in G_n induced by a strategy profile $\vec{M}$ in $\mathcal{G}$.

Definition 6.4.1 $\vec{M} = \{M_1, \ldots, M_c\}$ *is a* computational Nash equilibrium *of a computable uniform sequence $\mathcal{G}$ if, for all players $i \in [c]$ and for all polynomial-time strategies M_i' in $\mathcal{G}$ for player i, there exists a negligible function ϵ, such that for all n,*

$$\sum_{h \in H_n^T} \psi_{\vec{M}}^{G_n}(h) u_i(h) \geq \sum_{h \in H_n^T} \psi_{(M', \vec{M}_{-i})}^{G_n}(h) u_i(h) - \epsilon(n).$$

Our definition of computational NE is similar in spirit to that of Dodis, Halevi, and Rabin [14], although they formalize it by having the strategies depend on a security parameter and the utilities depend only on actions in a single normal-form game (rather than a sequence of extensive-form games). Our definition (and theirs) differs from the standard definition of ϵ-NE in two ways. First, we restrict to polynomial-time deviations. This seems in keeping with our focus on polynomial-time players. Second, we have a negligible loss of utility ϵ in the definition, and ϵ depends on the deviation. (The fact that ϵ depends on the deviation means that what we are considering cannot be considered an ϵ-Nash equilibrium in the standard sense.) Of course, we could have given a definition more in the spirit of the standard definition of Nash equilibrium by simply taking ϵ to be 0. However, the resulting solution concept would simply not be very interesting, given our restriction to polynomial-time players. In general, there will not be a "best" polynomial-time strategy; for every polynomial-time TM, there may be another TM that is better and runs only slightly longer. For example, player 2 may be able to do a little better by spending a little more time trying to decrypt the commitment in a commitment scheme. (See also the examples in [29].)

We now show that our model allows us to provide conditions that guarantee the existence of a computational NE; to the best of our knowledge, this has not been done before (and is mentioned as an open question in [45]). More specifically, we show that if a computational game $\mathcal{G}$ represents G, then for every NE $\vec{\sigma}$ in G, there is a corresponding NE in $\mathcal{G}$.

Theorem 6.4.2 *If $\mathcal{G}$ $\langle \vec{f}, \mathcal{F} \rangle$-represents G and $\vec{\sigma}$ is a NE in G, then $\mathcal{F}(\vec{\sigma})$ is a computational NE of $\mathcal{G}$.*

Proof: Suppose that $\vec{\sigma}$ is a NE in G. By UG4, $\vec{\sigma}$ corresponds to $\mathcal{F}(\vec{\sigma})$. Thus, there exists some negligible function ϵ such that, for all n,

$$\sum_{h \in H^T} \phi^{G_n}_{\mathcal{F}(\vec{\sigma})}(h) u_i(h) > \sum_{h \in H^T} \rho^G_{\vec{\sigma}}(h) u_i(h) - \epsilon(n).$$

We claim that $\vec{M}_{\vec{\sigma}}$ is a computational NE of $\mathcal{G}$. Assume, by way of contradiction, that it is not. That means there is some player i, some strategy M'_i for player i, and some constant $c > 0$ such that, for infinitely many values of n,

$$\sum_{h \in H^T} \phi^{G_n}_{(M', \mathcal{F}(\vec{\sigma}_{-i}))}(h) u_i(h) > \sum_{h \in H^T} \phi^{G_n}_{\mathcal{F}(\vec{\sigma})}(h) u_i(h) + \frac{1}{n^c};$$

If not, we could have constructed a negligible function to satisfy the equilibrium condition.

By combining the two equation we get that for infinitely many values of n,

$$\sum_{h \in H^T} \phi^{G_n}_{(M', \mathcal{F}(\vec{\sigma}_{-i}))}(h) u_i(h) > \sum_{h \in H^T} \rho^G_{\vec{\sigma}}(h) u_i(h) - \epsilon(n) + \frac{1}{n^c}.$$

Since $\vec{\sigma}$ is a NE, we get that for all sequences $\sigma'_1, \sigma'_2 \ldots$ of strategies for player i in G,

$$\sum_{h \in H^T} \rho^G_{\vec{\sigma}}(h) u_i(h) \geq \sum_{h \in H^T} \rho^G_{(\sigma'_n, \vec{\sigma}_{-i})}(h) u_i(h).$$

This means that for infinitely many values of n, and for any such sequence,

$$\sum_{h \in H^T} \phi^{G_n}_{(M', \mathcal{F}(\sigma_{-i}))}(h) u_i(h) > \sum_{h \in H^T} \rho^{G}_{(\sigma'_n, \vec{\sigma}_{-i})}(h) u_i(h) - \epsilon(n) + \frac{1}{n^c}.$$

But this contradicts UG4(c), which says that there must exist a sequence $\sigma'_1, \sigma'_2 \ldots$ such that $\{\phi^{G_n}_{(M'_i, \mathcal{F}(\vec{\sigma}_{-i}))}\}_n$ is computationally indistinguishable from $\{\rho^{G}_{(\sigma'_n, \vec{\sigma}_{-i})}\}_n$. Since the difference between the two payoffs is not negligible, a distinguisher could just sample enough outcomes of these strategies and compute the average payoff to distinguish the two distributions with non-negligible probability. Thus, $\vec{M}_{\vec{\sigma}}$ must be a computational NE of $\mathcal{G}$. □

Theorem 6.4.2 shows that every NE in G has a corresponding NE in $\mathcal{G}$. The converse does not hold. This should not be surprising; the set of strategies in $\mathcal{G}$ is much richer than that in G. The following example gives a simple illustration.

Example 6.4.3 Consider the 2-player game G' that is like the game in Figure 6.1, except that the payoff is 1 to both if they match and 0 otherwise (and both get -1 if player 1 does not open the envelope). This game has three NE: both play 0; both play 1; and both play the mixed strategy that gives probability $1/2$ to each of 0 and 1. There is a computational game $\mathcal{G}'$ that represents G' that is essentially identical to the game $\mathcal{G}$ described in Section 6.3.2, except that the payoffs are modified appropriately. The game $\mathcal{G}'$ has many more equilibria than G', since player 1 can commit to 0 and 1 with 0.5 probability but use a fixed key that the second player knows (or choose a random key from a low entropy set that the second player can enumerate). Player 2 can take advantage of this to always play the matching action. There is no strategy in G' that can mimic this behavior.

6.4.2 Computational sequential equilibrium

Our goal is to define a notion of computational sequential equilibrium. To do so, it is useful to think about the standard definition of sequential equilibrium at an abstract level. Essentially, $\vec{\sigma}$ is a sequential equilibrium if, for each player i, there is a partition $\mathcal{I}_i'$ of the histories where i plays such that, at each cell $I \in \mathcal{I}_i'$, player i has beliefs about the likelihood of being at each history in I, and the action that he chooses at a history in I according to σ_i is a best response, given these beliefs and what the other agents are doing (i.e., σ_{-i}). The standard definition of sequential equilibrium takes the partition $\mathcal{I}_i'$ to consist of i's information sets. If we partition the histories into singletons, we get a *subgame-perfect equilibrium* [63]. As we argued in Section 6.3.3, the information sets in $\mathcal{G}$ are too fine, in general, to capture a player's ability to distinguish. Thus, as a first step to getting a notion of computational sequential equilibrium, we generalize the standard definition of sequential equilibrium in a straightforward way to get $\vec{\mathcal{I}}$-sequential equilibrium, where $\mathcal{I}_i$ is an arbitrary partition of the histories where i plays.

Definition 6.4.4 *Given a partition $\vec{\mathcal{I}}$, $\vec{\sigma}$ is a $\vec{\mathcal{I}}$-*sequential equilibrium *of G if there exists a sequence of completely mixed strategy profiles $\vec{\sigma}^1, \vec{\sigma}^2, \ldots$ converging to $\vec{\sigma}$ and a sequence $\delta_1, \delta_2, \ldots$ of nonnegative real numbers converging to 0 such that, for each player i and each set $I \in \mathcal{I}_i$, $\vec{\sigma}_i^n$ is a δ_n-best response to $\vec{\sigma}_{-i}^n$ conditional on having reached I.*

What are reasonable partition structures to use when considering a computational game? As we suggested, using the information partition structure of $\mathcal{G}$ seems unreasonable. For example, in our example commitment game, this does

not allow the second player to act the same when facing commitments to 0 and commitments to 1, although, as we argued earlier, if player 1 plays appropriately, a computationally bounded player cannot distinguish these two events.

It seems reasonable to have histories in the same cell of the partition if the player cannot distinguish what these histories actually "represent". For general uniform computable sequences it is unclear what "represents" should mean. However, if $\mathcal{G}$ represents a game G, then we do have in some sense a representation for a history: the history it maps to in the underlying game. As we saw in Section 6.3.3, what a player can infer from a history might depend not just on the information partition structure of the games in $\mathcal{G}$, but also on the strategies played by the players in G. Thus, a natural candidate for a partition structure $\vec{I}$ when $\vec{M}$ is the strategy profile played is a partition that is based on an $\vec{M}$-consistent partition structure $\vec{\mathcal{I}}_G$ of the histories of G. We now formalize this intuition.

Suppose that $\mathcal{G}$ $\langle \vec{f}, \mathcal{F} \rangle$-represents G. Given a set $I \subseteq H$, let I_{G_n} be the set consisting of histories $h \in G_n$ such that $f_n(h) \in I$. Given two strategies M and M' for a player in $\mathcal{G}$, let (M, I, M') be the TM that plays like M in G_n up to I_{G_n}, and then switches to playing M' from that point on. For a game $G_n \in \mathcal{G}$, a strategy profile $\vec{M}$, and a set H'_n of histories in G_n that is reached with positive probability when $\vec{M}$ is played, let $\phi^{G_n}_{\vec{M},H'_n}$ be the probability on terminal histories in G induced by pushing forward the probability on terminal histories in G_n conditioned on reaching H'_n (where we identify the event "reaching H'_n" with the set of terminal histories that extend a history in H'_n). We can similarly define $\rho^{G}_{\vec{\sigma},H'}$ for a subset H' of histories in G.

Definition 6.4.5 *Suppose that $\mathcal{G}$ $\langle \vec{f}, \mathcal{F} \rangle$-represents G. Then $\vec{M} = \{M_1, \ldots, M_c\}$ is a*

computational sequential equilibrium *of $\mathcal{G}$ if there exists a sequence of completely mixed strategies $\vec{M}^1, \vec{M}^2, \ldots$ converging to $\vec{M}$ and a sequence $\delta_1, \delta_2, \ldots$ converging to 0 such that, for all k, n, and players $i \in [c]$, there exists an $\vec{M}$-consistent partition $\mathcal{I}_i$ such that, for all sets $I \in \mathcal{I}_i$ and all polynomial-time strategies M' for player i in $\mathcal{G}$, there exists a negligible function ϵ such that*

$$\sum_{h \in H^T} \phi^{G_n}_{\vec{M}^k, I_{G_n}}(h) u_i(h) \geq \sum_{h \in H^T} \phi^{G_n}_{((\vec{M}^k_i, I, M'), \vec{M}^k_{-i}), I_{G_n}}(h) u_i(h) - \epsilon(n) - \delta_k.$$

We now claim that, as with NE, if $\vec{\sigma}$ is a sequential equilibrium of an extensive form game G with perfect recall and $\mathcal{G}$ $\langle \vec{f}, \mathcal{F} \rangle$-represents G, then $\mathcal{F}(\vec{\sigma})$ is a computational sequential equilibrium of $\mathcal{G}$.

Theorem 6.4.6 *Suppose that $\mathcal{G}$ $\langle \vec{f}, \mathcal{F} \rangle$-represents G and G has perfect recall. If there exists a belief function μ such that $(\vec{\sigma}, \mu)$ is a sequential equilibrium in G, then $\mathcal{F}(\vec{\sigma})$ is a computational sequential equilibrium of $\mathcal{G}$.*

Proof: Suppose that there exists a belief system μ such that $(\vec{\sigma}, \mu)$ is a sequential equilibrium in G. Thus, there exists a sequence of completely mixed strategy profiles $\vec{\sigma}^1, \vec{\sigma}^2, \ldots$ that converges to $\vec{\sigma}$ and a sequence $\delta_1, \delta_2, \ldots$ that converges to 0 such that for all players i, all information sets I for i in G, and all strategies σ' for i that act like σ on all prefixes of histories in I, we have that

$$\sum_{h \in H^T} \rho^G_{\vec{\sigma}^k, I}(h) u_i(h) \geq \sum_{h \in H^T} \rho^G_{(\sigma', \vec{\sigma}^k_{-i}), I}(h) u_i(h) - \delta_k. \tag{6.1}$$

Assume, by way of contradiction, that $\vec{M} = \mathcal{F}(\vec{\sigma})$ is not a computational sequential equilibrium. Let M^k_i be the TM that acts like $\mathcal{F}(\sigma^k_i)$ except that at a view it is called to play, with probability $\frac{1}{2^{nk}}$ (which is negligible), it plays an arbitrary legal action, chosen uniformly at random. Note that this makes M^k_i completely

mixed, while ensuring that $\vec{M}^k$ still corresponds to $\vec{\sigma}^k$. Also note that the sequence $\vec{M}^1, \vec{M}^2, \ldots$ converges to $\vec{M}$. By Theorem 6.3.6, if $\mathcal{I}_i$ is the information partition of player i in G, then $\mathcal{I}_i$ is $\vec{M}^k$-consistent for all k, and, in particular, is also $\vec{M}$-consistent. Since $\vec{M}$ is not a computational sequential equilibrium, there must be some k, player i, information set I for i in G, strategy M_i' for i, and constant c such that, for infinitely many values of n,

$$\sum_{h \in H^T} \phi^{G_n}_{((\vec{M}_i^k, I, M'), \vec{M}_{-i}^k), I_{G_n}}(h) u_i(h) > \sum_{h \in H^T} \phi^{G_n}_{\vec{M}^k, I_{G_n}}(h) u_i(h) + \frac{1}{n^c} + \delta_k. \tag{6.2}$$

Since $\vec{\sigma}^k$ is completely mixed, every terminal history is reached with positive probability. Thus, I_{G_n} is reached with positive probability. Since $\vec{M}^k$ corresponds to $\vec{\sigma}^k$, $\{\phi^{G_n}_{\vec{M}^k, I_{G_n}}\}_n$ (the conditional ensemble) must be statistically close to $\{\rho^G_{\vec{\sigma}^k, I}\}_n$, for otherwise we could use the distinguisher for these ensembles to distinguish the unconditional ensembles. It follows that there exists some negligible function ϵ such that, for all n,

$$\sum_{h \in H^T} \phi^{G_n}_{\vec{M}^k, I_{G_n}}(h) u_i(h) > \sum_{h \in H^T} \rho^G_{\vec{\sigma}^k, I}(h) u_i(h) - \epsilon(n). \tag{6.3}$$

From (6.2) and (6.3), it follows that, for infinitely many values of n,

$$\sum_{h \in H^T} \phi^{G_n}_{((\vec{M}_i^k, I, M'), \vec{M}_{-i}^k), I_{G_n}}(h) u_i(h) > \sum_{h \in H^T} \rho^G_{\vec{\sigma}^k, I}(h) u_i(h) - \epsilon(n) + \frac{1}{n^c} + \delta_k. \tag{6.4}$$

By UG4(c), there is a sequence $\sigma_1', \sigma_2', \ldots$ of strategies for i in G such that $\{\phi^{G_n}_{((\vec{M}_i^k, I, M'), \vec{M}_{-i}^k)}\}_n$ is computationally indistinguishable from $\{\rho^G_{(\sigma_n', \vec{\sigma}_{-i}^k)}\}_n$. Since, for n sufficiently large, I_{G_n} is reached with non-negligible probability by $\vec{M}^k$, and $(\vec{M}_i^k, I, M')$ acts like $\vec{M}_i^k$ in all prefixes of histories in I_{G_n}, it must be the case that for n sufficiently large, $((\vec{M}_i^k, I, M'), \vec{M}_{-i}^k)$ reaches I_{G_n} with non-negligible probability. Moreover, $\{\phi^{G_n}_{((\vec{M}_i^k, I, M'), \vec{M}_{-i}^k), I_{G_n}}\}_n$ is computationally indistinguishable from $\{\rho^G_{(\sigma_n', \vec{\sigma}_{-i}^k), I}\}_n$. If not, again, a distinguisher for the unconditional distributions can just use the distinguisher for the conditional distribution by calling

it only when the sampled history is such that I is visited. From (6.1) and (6.4), we get that for infinitely many values of n,

$$\sum_{h \in H^T} \phi^{G_n}_{((\vec{M}_i^k, \mathcal{I}(I), M'), \vec{M}_{-i}^k), I_{G_n}}(h) u_i(h) > \sum_{h \in H^T} \rho^{G}_{(\sigma'_n, \vec{\sigma}_{-i}^k), I}(h) u_i(h)(h) u_i(h) - \epsilon(n) + \frac{1}{n^c}.$$

But, as in previous arguments, this contradicts the assumption that $\{\phi^{G_n}_{((\vec{M}_i^k, I, M'_i), \vec{M}_{-i}^k), I_{G_n}}\}_n$ is computationally indistinguishable from $\{\rho^{G}_{(\sigma'_n, \vec{\sigma}_{-i}^k), I}\}_n$. Thus, $\vec{M}_{\vec{\sigma}}$ is a computational sequential equilibrium of $\mathcal{G}$. □

What are the beliefs represented by this equilibrium? The beliefs we get are such that the players believe that, except with negligible probability, only strategies that are mappings (via $\mathcal{F}$) of strategies in the underlying game were used, so they explain deviations in the computational game in terms of deviations in the underlying game.

One consequence of using completely mixed strategies in the standard setting is that a player always assigns positive probability to wherever he may find himself. In our setting, while we also require strategies to be completely mixed, a player i may still find himself in a situation (i.e., may have a view) to which he ascribes probability 0, so he knows his beliefs are bound to be incorrect. This can happen only if the randomness in i's state is inconsistent with the moves that i made that led to the current view. (This can happen if, for example, i ignored the random string when computing the commitment string, and just outputted a string of all 1's.) While i may ascribe probability 0 to his earlier moves, deviations by other players always result in views to which i ascribes positive probability, so such deviations cannot be used as signals or threats.

By considering a consistent partitions here, we effectively average the expected payoff over all histories of $\mathcal{G}_n$ that map to the same information set in

I. Note that, for each specific history in this set, there might be a better TM. For example, in the commitment game discussed before, for each commitment string, there is a TM for player 2 that does better than the prescribed protocol: the one that plays the right value given that string. However, our notion considers the expected value over all these histories, and thus a good deviation does not exist. Since no polynomial-time TM can tell to which histories in the underlying game these histories are mapped (via f), we treat cells in a consistent partition just as traditional information sets are treated in the standard notion of sequential equilibrium.

6.5 Application: Implementing a Correlated Equilibrium Without a Mediator

In this section, we show that our approach can help us analyze protocols that use cryptography to implement a correlated equilibrium (CE) in a normal-form game. Dodis, Halevi, and Rabin [14] (DHR) were the first to use cryptographic techniques to implement a CE. They did so using a protocol that they showed was a NE, provided that players are computationally bounded (for a notion of computational NE that is related to ours). However, as discussed by Gradwohl, Livne, and Rosen [26] (GLR), DHR's proposed protocol does not satisfy solution concepts that also require some sort of sequential rationality. DHR's protocol punishes deviations using a minimax strategy that may give the punisher as well as the player being punished a worse payoff; thus, it is just an empty threat. To deal with this issue, GLR introduce a solution concept that they call *Threat Free Equilibrium (TFE)*, which explicitly eliminates such empty threats.

GLR additionally provide a protocol that can implement a CE in a normal-form game that is a convex combination of NEs (CCNE), without using a mediator; the GLR protocol is a TFE if the players are computationally bounded.

We now provide a protocol similar in spirit to the one used in GLR that implements a CCNE; our protocol is a computational sequential equilibrium if the players are computationally bounded. Unlike GLR, we are able to apply our approach to CEs in games with more than 2 players, as well as being able to implement CCNEs that are not Pareto optimal. One more advantage of our approach is that since we allow the underlying game to be one of imperfect information, there is a natural way to model a normal-form game (where players are assumed to move simultaneously) as an extensive-form game: players just move sequentially without learning what the other player does. Since GLR's results apply only to games of perfect information, they had to argue that they could extend their result to normal-form games.

We require that the CCNE is of finite support, that all its coefficients are rational numbers, and that each of the NEs in its support has coefficients that are rational numbers.[5] We call such CCNEs *nice*. Note that any CCNE can be approximated to arbitrary accuracy by a nice CCNE.

Given a normal-form game G with a nice CCNE π, we show how to convert it to an extensive-form game G_{corr} that implements this CE without using cryptography, but using envelopes; that is, G_{corr} has a sequential equilibrium with the same distribution over outcomes in G as π. We then show how to implement G_{corr} as a computational game using a cryptographic protocol.

[5]GLR also made these assumptions. In fact, they required a slightly stronger condition; they required all the coefficients to be rational numbers whose denominator is a power of two.

Given G and π, let ℓ be the least common denominator of the coefficients of π. Let G_{corr} be the game where player 1 first puts an element of $\{0, \ldots, \ell - 1\}$ in an envelope, then player 2 plays an element in $\{0, \ldots, \ell - 1\}$ without knowing what player 1 played (all the histories where player 2 makes his first move are in the same information set of player 2). Then player 1 can either open the envelope or destroy it. All the histories after player 1 opens the envelope form singleton information sets for the other players; all histories after player 1 destroys the envelope and 2 initially played j are in the same information set for the players other than 1, for $j \in \{0, \ldots, \ell - 1\}$. Then G is played. (Note that G might involve many players other than 1 and 2, but 1 and 2 are the only players who play in the initial part of G_{corr}.) The players move sequentially: first player 2 moves, then player 1 moves (without knowing player 2's move), then player 3 moves (without knowing 1 and 2's moves), and so on. The payoffs of G_{corr} depend only on the players' moves when playing the G component of G_{corr}, and are the same as the payoffs in G. See Figure 6.3 for a game G_{corr} when ℓ is 2 and G is a coordination game: that is, in G, each player moves either left or right, and each gets a payoff of 1 if they make the same move, and -1 if they make different moves.

Let σ be a NE in G in which player 1's payoff is no better than it is in any other NE in G. Now consider the following simple strategies for the players in G_{corr}. Intuitively, the players start by picking a NE in the support of π to play, with probability proportional to its coefficient in π. To this end, fix an ordering of length ℓ of the NEs in the support of π, where each NE appears a number of times proportional to its weight in the convex combination that makes up π. At the empty history, player 1 selects an action a uniformly at random from $\{0, \ldots, \ell - 1\}$ and puts it in the envelope. Then player 2 also selects an action

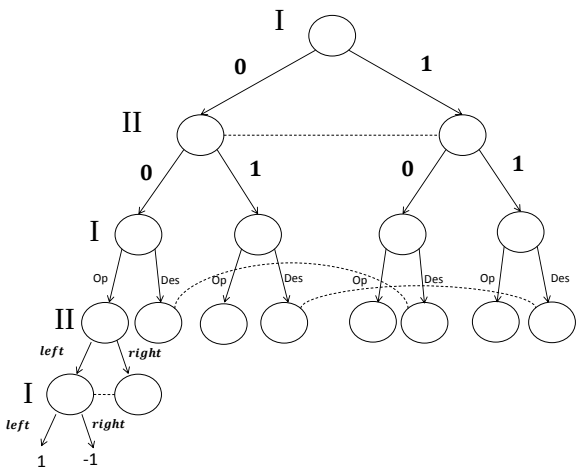

Figure 6.3: An example of the game G_{corr} where $\ell = 2$ and G is a coordination game

b uniformly at random from $\{0, \ldots, \ell - 1\}$. Then player 1 opens the envelope. The players then play the NE in place $(a + b \mod \ell)$ in the ordering of NEs. If player 1 does not open, the players play according to σ. Call the resulting strategy profile $\vec{\sigma}_\pi$. It is not hard to verify that $\vec{\sigma}_\pi$ implements π, and that there exists a probability measure μ such that $(\vec{\sigma}_\pi, \mu)$ is a sequential equilibrium of G_{corr}. Defining μ is easy: the only information sets not reached with positive probability (and hence μ is determined) are the one where "destroy" is played. At that point, the players' play σ, so they are best responding to each other, no matter what their beliefs are.

So now all we have to provide is a computational game $\mathcal{G}_{corr}$ that represents G_{corr}, where the games in $\mathcal{G}_{corr}$ use cryptography instead an envelope for the first part of the game. Let d be such that $2^{d-1} \leq \ell < 2^d$. Let $\mathcal{G}_{corr}$ be the sequence where G_n is the game where, at the empty history, player 1 commits to a d-bit string by using d commitments in parallel, each with key length $n - 1$ and outputs the d commitment strings as his action. Player 2 then plays a bitstring of

length d that can be viewed as a binary representation of a number in $\{0, \ldots, \ell - 1\}$. Player 1 then plays a string that is intended to be the commitment keys of the d commitments. Then the players play a string representing their action in G (again using its binary representation). The utility are then given by the utility functions in G.

We now claim that $\mathcal{G}_{corr}$ represents G_{corr}.

Theorem 6.5.1 *$\mathcal{G}_{corr}$ represents G_{corr}.*

Proof: It is obvious that $\mathcal{G}_{corr}$ is a computable uniform sequence. We now show that it represents G_{corr}. The mappings $\vec{f}$ of histories maps player 1's commitments to a string s to the action $s \mod \ell$. (Notice that the fact we used d commitments in parallel does not change the fact that the commitments are perfectly binding and thus this is well defined.) Actions of player 2 are mapped to an action $s \mod \ell$ according to their binary representation; if player 1 reveals d valid keys in h, then in $f_n(h)$ he plays "open", and otherwise he plays "destroy"; the actions of G are mapped in the obvious way.

To show that UG4 holds, we proceed as follows: The mapping $\mathcal{F}$ for a player j other than 1 and 2 is obvious: It is easy to compute using the TM R of the commitment scheme if the commitments were opened successfully or not, so j can compute at which information set of G_{corr} he is at (given his view), and play the binary representation of the action that the strategy plays at that information set. For player 2, note that player 2's first action in G_{corr} can't depend on player 1's action, since player 2's information set contains all the histories. Thus, a deterministic strategy σ_2 for player 2 in G_{corr} just plays an action in $\{0, \ldots, \ell - 1\}$; $\mathcal{F}(\sigma_2)$ just plays the same action at player 2's first information set in $\mathcal{G}_{corr}$.

Similarly to the other players, $\mathcal{F}(\sigma_2)$ also plays the same action in G as σ_2 when player 2 is called upon to play again. Given a deterministic strategy σ_1 for player 1, if σ_1 plays a at the first step in $\mathcal{G}_{corr}$, $\mathcal{F}(\sigma_1)$ chooses uniformly at random one of the d-bit strings such that $s = a \mod \ell$ (there are at most 2 such strings), and plays the commitments strings $C_1(1^n, s_1, r_1), \ldots, C_d(1^n, s_d, r_d)$, where $r = r_1 || \ldots || r_d$ is the prefix of the random tape representing the randomness used to compute the commitment strings. To play the action "open", it computes $k_i = C_2(1^n, s_i, r_i)$ and play $k_1 || \ldots || k_d$; to play "destroy", it plays $k_1 || \ldots || k_d \oplus 1$ (a string other than the right keys). Again, it is obvious how player 1 plays in G. It is easy to see that $\mathcal{F}(\vec{\sigma})$ corresponds to $\vec{\sigma}$, so UG4(a) holds. UG4(b) holds for all players trivially given these strategies.

It is also obvious that UG4(c) holds for player 1. Since the information structure it faces at $\mathcal{G}_{corr}$ and G_{corr} is essentially the same, anything it can do in $\mathcal{G}_{corr}$ can be done by a strategy in G_{corr} by just looking at the distribution of actions in histories that map to each information set.

The other players have different information structures in $\mathcal{G}_{corr}$ and G_{corr}, since they see the commitment strings in $\mathcal{G}_{corr}$. We discuss UG4(c) for player 2 here; the argument in the case of the others is similar (and simpler). Let σ_i for $i \neq 2$ be a strategy for player i in G_{corr}, and let $M_i = \mathcal{F}(\sigma_i)$. Let M' be an arbitrary polynomial time strategy for player 2 in $\mathcal{G}_{corr}$, and let D_1^n be the distribution M''s first action in G_n; let $D_{j,w}^n$ be the distribution over the actions of M' in G given that the commitment was opened successfully, player 1 committed to j, and player 2's first move was w; and let D_w^n be the distribution over the actions of M' in G_n if the commitment is not opened successfully and player 2's first move was w. Let σ'_n be a strategy in G_{corr} for player 2 that plays according

to these distributions. We claim that $\{\phi^{G_n}_{(M_1,M',\ldots,M_c)}\}_n$ is indistinguishable from $\{\rho^{G_{corr}}_{(\sigma_1,\sigma'_n,\ldots,\sigma_c)}\}_n$.

Let $\phi^{G_n,1}_{(M_1,M',\ldots,M_c)}$ be the distribution over histories ending at the first action of player 2 when $(M_1, M', \ldots, M_c)$ is played in G_n and mapped using f_n to histories of G_{corr}, and let $\rho^{G_{corr},1}_{(\sigma_1,\sigma'_n,\ldots,\sigma_c)}$ be the distribution over partial histories ending at the first action of player 2 when $(\sigma_1, \sigma'_n, \ldots, \sigma_c)$ is played in G_{corr}. We first claim that $\{\phi^{G_n,1}_{(M_1,M',\ldots,M_c)}\}_n$ is indistinguishable from $\{\rho^{G_{corr},1}_{(\sigma_1,\sigma'_n,\ldots,\sigma_c)}\}_n$. Assume, by way of contradiction, that it is not. This can happen only if, for infinitely many n, M' plays some action a with probabilities that differ non-negligibly, depending on whether it is faced with a commitment to different strings s or s'. But that means that for infinitely many n, it can distinguish those two events with non-negligible probability. This contradicts the assumption that the commitment scheme is secure. (Note that it is easy to show that, because a single commitment has the hiding property, then even when d such commitments are run in parallel, no polynomial-time TM should be able to distinguish between commitments to s and s'.)

It is easy to see that this also means that the distribution over partial histories just before player 2 plays again are also indistinguishable. Now if the commitment is opened successfully, then the information structure player 2 faces in $\mathcal{G}_{corr}$ is the same as in G_{corr}, and thus the statement is obviously true. If the commitments were not opened, than by using a argument similar to that used for player 2's first action, we can argue that if the distributions over partial histories just after player 2 plays again are not indistinguishable, then again we can use that as a distinguisher for the commitment scheme. □

By Theorems 6.4.6 and 6.5.1, since $\vec{\sigma}_\mu$ (with the appropriate beliefs) is a se-

quential equilibrium of G_{corr}, $\mathcal{F}(\vec{\sigma}_\mu)$ is a computational sequential equilibrium of $\mathcal{G}_{corr}$.

BIBLIOGRAPHY

[1] Garrett Andersen and Vincent Conitzer. Fast equilibrium computation for infinitely repeated games. In *Proc. of The 27th AAAI Conference on Artificial Intelligence*, 2013.

[2] Robert J. Aumann and Lloyd S. Shapley. Long-term competitiona game-theoretic analysis. In Nimrod Megiddo, editor, *Essays in Game Theory*, pages 1–15. Springer New York, 1994.

[3] Elchanan Ben-Porath. Repeated games with finite automata. *Journal of Economic Theory*, 59:17–32, 1993.

[4] Eli Ben-sasson, Ehud Kalai, and Adam Kalai. An approach to bounded rationality. In *Advances in Neural Information Processing Systems 19*, pages 145–152. MIT Press, 2007.

[5] Guido Biele, Ido Erev, and Eyal Ert. Learning, risk attitude and hot stoves in restless bandit problems. *Journal of Mathematical Psychology*, 53(3):155–167, 2009.

[6] David Blackwell. Comparison of experiments. In *Second Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 93–102, 1951.

[7] David Blackwell. Equivalent comparisons of experiments. *The Annals of Mathematical Statistics*, pages 265–272, 1953.

[8] Christian Borgs, Jennifer Chayes, Nicole Immorlica, Adam Tauman Kalai, Vahab Mirrokni, and Christos H. Papadimitriou. The myth of the folk theorem. *Games and Economic Behavior*, 70(1):34–43, 2010.

[9] Wei Chen, Shu-Yu Liu, Chih-Han Chen, and Yi-Shan Lee. Bounded memory, inertia, sampling and weighting model for market entry games. *Games*, 2(1):187–199, 2011.

[10] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Computing Nash equilibria: Approximation and smoothed complexity. In *Proc. of the 47th IEEE Symposium on Foundations of Computer Science*, pages 603–612, 2006.

[11] Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player nash equilibria. *Journal of the ACM*, 56(3):14:1–14:57, May 2009.

[12] Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259, 2009.

[13] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976.

[14] Yevgeniy Dodis, Shai Halevi, and Tal Rabin. A cryptographic solution to a game theoretic problem. In *Advances in Cryptology–CRYPTO 2000*, pages 112–130, 2000.

[15] Ido Erev, Eyal Ert, and Alvin E. Roth. A choice prediction competition for market entry games: An introduction. *Games*, 1:117–136, 2010.

[16] Ido Erev, Eyal Ert, and Alvin E. Roth. Market entry prediction competition web site. Available online at https://sites.google.com/site/gpredcomp/, 2010.

[17] Ido Erev, Eyal Ert, Alvin E. Roth, Ernan Haruvy, Stefan M. Herzog, Robin Hau, Ralph Hertwig, Terrence Stewart, Robert West, and Christian Lebiere. A choice prediction competition: Choices from experience and from description. *Journal of Behavioral Decision Making*, 23(1):15–47, 2010.

[18] Drew Fudenberg and Eric Maskin. The folk theorem in repeated games with discounting or with incomplete information. *Econometrica*, 54(3):533–554, 1986.

[19] Itzhak Gilboa and Dov Samet. Bounded versus unbounded rationality: The tyranny of the weak. *Games and Economic Behavior*, 1(3):213–221, 1989.

[20] Oded Goldreich. *Foundation of Cryptography, Volume I Basic Tools*. 2001.

[21] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, 33(4):792–807, 1986.

[22] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.

[23] Irving J. Good. Rational decisions. *Journal of the Royal Statistical Society. Series B (Methodological)*, 14(1), 1952.

[24] Olivier Gossner. *Repeated games played by cryptographically sophisticated players*. Center for Operations Research & Econometrics. Université catholique de Louvain, 1998.

[25] Olivier Gossner. Sharing a long secret in a few public words. Technical report, THEMA (THéorie Economique, Modélisation et Applications), Université de Cergy-Pontoise, 2000.

[26] Ronen Gradwohl, Noam Livne, and Alon Rosen. Sequential rationality in cryptographic protocols. *ACM Transactions on Economics and Computation*, 1(1):2:1–2:38, January 2013.

[27] Joseph Y. Halpern and Rafael Pass. Game theory with costly computation. In *Proc. of the First Symposium on Innovations in Computer Science*, pages 120–142, 2010.

[28] Joseph Y. Halpern and Rafael Pass. Sequential equilibrium in computational games. In *Proc. of the 23rd International Joint Conference on Artificial Intelligence*, pages 171–176, 2013.

[29] Joseph Y. Halpern, Rafael Pass, and Daniel Reichman. On the nonexistence of equilibrium in computational games. 2015.

[30] Joseph Y. Halpern, Rafael Pass, and Lior Seeman. I'm doing as well as I can: Modeling people as rational finite automata. In *Proc. of the 26th AAAI Conference on Artificial Intelligence*, 2012.

[31] Joseph Y. Halpern, Rafael Pass, and Lior Seeman. Decision theory with resource-bounded agents. *Topics in Cognitive Science*, 6(2):245–257, 2014.

[32] Joseph Y. Halpern, Rafael Pass, and Lior Seeman. The truth behind the myth of the folk theorem. In *Proc. of the 5th Conference on Innovations in Theoretical Computer Science*, pages 543–554, 2014.

[33] Joseph Y. Halpern, Rafael Pass, and Lior Seeman. Not just an empty threat: Subgame-perfect equilibrium in repeated games played by computationally bounded players. In *Proc. of the 10th Conference on Web and Internet Economics (WINE)*, 2014.

[34] Joseph Y. Halpern, Rafael Pass, and Lior Seeman. Computational extensive-form games. *arXiv preprint arXiv:1506.03030*, 2015.

[35] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM Journal on Computing*, 28(4):1364–1396, 1999.

[36] Martin E. Hellman and Thomas M. Cover. Learning with finite memory. *The Annals of Mathematical Statistics*, 41(3):765–782, 1970.

[37] Jack Hirshleifer. The private and social value of information and the reward to inventive activity. *The American Economic Review*, 61(4):561–574, 1971.

[38] Wassily Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30, 1963.

[39] Pavel Hubáček, Jesper B. Nielsen, and Alon Rosen. Limits on the power of cryptographic cheap talk. In *Advances in Cryptology–CRYPTO 2013*, pages 277–297, 2013.

[40] Pavel Hubáček and Sunoo Park. Cryptographically blinded games: leveraging players' limitations for equilibria and profit. In *Proc. of the 15th ACM Conference on Economics and Computation*, pages 207–208, 2014.

[41] Albert Xin Jiang and Kevin Leyton-Brown. Polynomial-time computation of exact correlated equilibrium in compact games. *Games and Economic Behavior*, 91:347 – 359, 2015.

[42] Daniel Kahneman and Amos Tversky. Prospect theory: an analysis of decision under risk. *Econometrica*, 47(2):263–292, 1979.

[43] Ehud Kalai. Bounded rationality and strategic complexity in repeated games. *Game Theory and Applications*, pages 131–158, 1990.

[44] Morton I. Kamien, Yair Tauman, and Shmuel Zamir. On the value of information in a strategic conflict. *Games and Economic Behavior*, 2(2):129–153, 1990.

[45] Jonathan Katz. Bridging game theory and cryptography: Recent results and future directions. In Ran Canetti, editor, *Theory of Cryptography*, volume 4948 of *Lecture Notes in Computer Science*, pages 251–272. Springer Berlin Heidelberg, 2008.

[46] Michael Kearns, Michael L. Littman, and Satinder Singh. Graphical models

for game theory. In *Proc. of the 7th Conference on Uncertainty in Artificial Intelligence*, pages 253–260, 2001.

[47] Gillat Kol and Moni Naor. Games for exchanging information. In *Proc. of the 40th Annual ACM Symposium on Theory of Computing*, pages 423–432, 2008.

[48] David M. Kreps and Robert Wilson. Sequential equilibria. *Econometrica*, 50(4):863–894, 1982.

[49] Ehud Lehrer. Internal correlation in repeated games. *International Journal of Game Theory*, 19(4):431–456, 1991.

[50] Michael L. Littman and Peter Stone. A polynomial-time Nash equilibrium algorithm for repeated games. *Decision Support Systems*, 39(1):55–66, 2005.

[51] John M. McNamara, Pete C. Trimmer, and Alasdair I. Houston. The ecological rationality of state-dependent valuation. *Psychological review*, 119(1):114, 2012.

[52] Nimrod Megiddo and Avi Wigderson. On play by means of computing machines. In *Proc. of the 1986 Conference on Theoretical Aspects of Reasoning About Knowledge*, pages 259–274, 1986.

[53] Roger B. Myerson. Game theory: analysis of conflict. *Harvard University*, 1991.

[54] Abraham Neyman. Bounded complexity justifies cooperation in the finitely repeated prisoners' dilemma. *Economics Letters*, 19(3):227–229, 1985.

[55] Abraham Neyman. The positive value of information. *Games and Economic Behavior*, 3(3):350–355, 1991.

[56] Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory*. MIT Press, Cambridge, Mass., 1994.

[57] Martin Peterson. *An introduction to decision theory*. Cambridge University Press, 2009.

[58] Sidney I. Resnick. *Adventures in Stochastic Processes*. Birkhauser, 1992.

[59] Ronald L. Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[60] Ariel Rubinstein. Equilibrium in supergames with the overtaking criterion. *Journal of Economic Theory*, 21(1):1–9, 1979.

[61] Ariel Rubinstein. Finite automata play the repeated prisoner's dilemma. *Journal of Economic Theory*, 39(1):83–96, 1986.

[62] Lior Seeman. I'd rather stay stupid: The advantage of having low utility. *arXiv preprint arXiv:1312.4187*, 2011.

[63] Reinhard Selten. Spieltheoretische behandlung eines oligopolmodells mit nachfrageträgheit. *Zeitschrift für Gesamte Staatswissenschaft*, 121:301–324 and 667–689, 1965.

[64] Herbert A. Simon. A behavioral model of rational choice. *The quarterly journal of economics*, 69(1):99–118, 1955.

[65] Michael Spence. Job market signaling. *The Quarterly Journal of Economics*, 87(3):355, 1973.

[66] Amparo Urbano and Jose E. Vila. Computational complexity and communication: Coordination in two-player games. *Econometrica*, 70(5):1893–1927, 2002.

[67] Amparo Urbano and Jose E. Vila. Unmediated communication in repeated games with imperfect monitoring. *Games and Economic Behavior*, 46(1):143–173, 2004.

[68] Peter Whittle. Restless bandits: Activity allocation in a changing world. *Journal of Applied Probability*, pages 287–298, 1988.

[69] Andrea Wilson. Bounded memory and biases in information processing. *Econometrica*, 82(6):2257–2294, 2014.